SQL TREINO

Aluno: joão Victor lopes Rodrigues
linkedin: https://www.linkedin.com/in/v%C3%ADctor-rodrigues-600151262/
Email: victorrodrigues9463@gmail.com

Exercícios de Consulta:

1.Liste todos os alunos.

```
select * from alunos;
```

2.Liste todos os cursos disponíveis.

```
select * from cursos;
```

3.Encontre o nome do curso que o aluno com o ID 5 está assistindo.

```
select alunos.id,alunos.nome,cursos.Nome
from alunos
inner join assistem
on alunos.id=assistem.idalunos
inner join cursos
on cursos.idcurso=assistem.idcursos
where alunos.id = 5;
```

4.Liste todos os alunos inscritos no curso com o ID 3.

```
select alunos.id,alunos.nome,cursos.Nome
from alunos
inner join assistem
on alunos.id=assistem.idalunos
inner join cursos
on cursos.idcurso=assistem.idcursos
where cursos.idcursos=3;;
```

5.Liste todos os alunos que nasceram nos anos 1/1/2000 e 31/12/2003;.

```
SELECT  nascimento
FROM alunos
WHERE nascimento BETWEEN '2000-01-01' AND '2003-12-31';
```

6.Encontre o nome do aluno que está assistindo ao curso "python".

```
select alunos.id,alunos.nome,cursos.Nome
from alunos
inner join assistem
on alunos.id=assistem.idalunos
inner join cursos
on cursos.idcurso=assistem.idcursos
where cursos.nome = ‘python’;;
```

7.Mostre os nomes dos alunos e os nomes dos cursos que estão assistindo.

```
select alunos.id,alunos.nome,cursos.Nome
from alunos
inner join assistem
on alunos.id=assistem.idalunos
inner join cursos
on cursos.idcurso=assistem.idcursos
```

8..Liste os cursos que não têm alunos inscritos.

```
SELECT cursos.idcurso, cursos.Nome
FROM cursos
LEFT JOIN assistem
ON cursos.idcurso = assistem.idcursos
WHERE assistem.idalunos IS NULL;
```

9.Encontre o nome do aluno que assiste o curso com o ID 4.

```
select alunos.id,alunos.nome,cursos.Nome
from alunos
inner join assistem
on alunos.id=assistem.idalunos
inner join cursos
on cursos.idcurso=assistem.idcursos
where cursos.idcursos=4;;
```

Exercícios de Inserção:

10.Adicione um novo aluno à tabela de alunos.

```
 insert into alunos value
(default,'Andre paulo','Redes','1985-12-05','M',65.00,1.64,'Brasil',19)
```

11.Insira um novo curso na tabela de cursos.

```
insert into cursos value
('33','Cotlin','Curso de cotlin completo !','80','120','2023');
```

12.Registre a inscrição do aluno com o ID 8 no curso com o ID 6.

```
insert into assistem
VALUES ('31', '2023-05-11', '10', '15');
```

13.Insira uma nova relação de aluno e curso na tabela "assimtem"

```
insert into assistem
VALUES ('31', '2023-05-11', '10', '15');
```

Exercícios de Atualização:

14.Atualize a idade do aluno com o ID 12 para 22 anos

```
update alunos
set nascimento = '2005-10-12'
where id = 12;
```

.

15.Mude o nome do curso com o ID 7 para "Introdução à Programação".

```
update cursos
set descricao = 'Curso de Introdução à Programação', Nome = 'Introdução à
Programação'
where idcurso = 7;
```

16.Atualize o nome do aluno com o ID 25 para "Joana Silva".

```
update alunos
set nome='Joana Silva'
where id= 25;
```

Exercícios de Exclusão:

17.Remova o aluno com o ID 18 da tabela de alunos

```
delete FROM alunos
where id = 18;.
```

18.Delete o curso com o ID 9 da tabela de cursos.

```
delete FROM cursos
where idcurso = 9;
```

19.Remova a inscrição do aluno com o ID 15 no curso com o ID 2.

```
DELETE FROM assistem
WHERE idalunos = 15 AND idcursos = 2;
```

Exercícios de Junção (JOIN):

20..Liste os alunos e os nomes dos cursos que eles estão assistindo.

```
select alunos.nome, cursos.Nome
from cursos
join assistem
on cursos.idcurso = assistem.idcursos
join alunos
on alunos.id = assistem.idalunos
```

21.Mostre os cursos e os nomes com carga horária maior que 30..

```
select alunos.nome, cursos.Nome
from cursos
join assistem
on cursos.idcurso = assistem.idcursos
join alunos
on alunos.id = assistem.idalunos
where cursos.carga>30;
```

22.Liste os alunos que estão assistindo a cursos com o ID 3.

```
select alunos.nome, cursos.Nome
from cursos
join assistem
on cursos.idcurso = assistem.idcursos
join alunos
on alunos.id = assistem.idalunos
where cursos.idcurso = 3;
```

Exercícios de Agregação:

23.Calcule o número médio de alunos por curso.

```
SELECT cursos.idcurso, cursos.Nome, AVG(assistem.idalunos) AS
media_alunos_por_curso
FROM cursos
JOIN assistem
 ON cursos.idcurso = assistem.idcursos
```

```
GROUP BY cursos.idcurso, cursos.Nome;
```

23.Encontre o nome do curso com o maior número de alunos inscritos.

```
SELECT cuesos.nome
FROM cursos
JOIN assintem
ON cursos.id = assintem_idcursos
GROUP BY cursos.nome
ORDER BY COUNT(*) DESC
LIMIT 1;
```

24.Calcule a idade média dos alunos inscritos no curso com o ID 5.

```
select  alunos.nome, AVG(alunos.idade), cursos.nome
from alunos
inner join assistem
on alunos.id = assistem.idalunos
inner join cursos
on cursos.idcursos = assistem.idcursos
where curso.idcursos = 5;
```

Exercícios de Subconsulta:

25.Liste os cursos que têm mais alunos inscritos do que o curso com o ID 3.

```
SELECT cursos.idcurso, cursos.nome, COUNT(assistem.idalunos) AS
quantidade_alunos
FROM cursos
INNER JOIN assistem
ON cursos.idcurso = assistem.idcursos
WHERE cursos.idcurso != 3
GROUP BY cursos.idcurso, cursos.nome
HAVING quantidade_alunos > (
   SELECT COUNT(idalunos)
   FROM assistem
   WHERE idcursos = 3)
```

26.Encontre os alunos que não estão inscritos em nenhum curso

```
SELECT alunos.id, alunos.nome
FROM alunos
LEFT JOIN assistem
ON alunos.id = assistem.idcursos
WHERE assistem.idalunos IS NULL;
```

.

Exercícios de Ordenação e Limite:

27.Liste os cinco primeiros alunos em ordem alfabética.

```
select * from alunos
group by nome
limit 5;
```

28.Mostre os três cursos com mais alunos inscritos.

```
SELECT cursos.idcurso, cursos.nome, COUNT(assistem.idalunos) AS quantidade_alunos
FROM cursos
INNER JOIN assistem ON cursos.idcurso = assistem.idcursos
GROUP BY cursos.idcurso, cursos.nome
ORDER BY quantidade_alunos DESC
LIMIT 3;
```

Exercícios de Filtros:

29.Mostre os cursos que têm a palavra "Avançado" em seus nomes.

```
select *
from cursos
where Nome LIKE'%Avançado'
```

Exercícios de União:

30.Liste os alunos que estão assistindo ao curso com o ID 10 ou ao curso com o ID 20.

```
select alunos.nome, cursos.Nome
from cursos
join assistem
on cursos.idcurso = assistem.idcursos
join alunos
on alunos.id = assistem.idalunos
where cursos.idcurso = 10 or cursos.idcurso=20;
```

Exercícios de Contagem:

31.Conte quantos alunos estão inscritos em cada curso

```
SELECT cursos.idcurso, cursos.nome, COUNT(assistem.idaluno) AS quantidade_alunos
FROM cursos
INNER JOIN assistem ON cursos.idcurso = assistem.idcursos
GROUP BY cursos.idcurso, cursos.nome
ORDER BY quantidade_alunos ;
```

.
32.Determine quantos cursos um aluno específico está assistindo.

```
select alunos.id, alunos.Nome, count(assistem.idcursos) as
cursos_total
from alunos
join assistem
on alunos.id = assistem.idalunos
group by alunos.id,alunos.Nome
order by cursos_total
```

Exercícios de Junção Externa:

33.Liste todos os cursos e, se houver, os alunos inscritos em cada um.

```
SELECT cursos.idcurso, cursos.nome AS nome_curso, alunos.id AS id_aluno, alunos.nome
AS nome_aluno
FROM cursos
LEFT JOIN assistem
ON cursos.idcurso = assistem.idcursos
LEFT JOIN alunos
ON assistem.idalunos = alunos.id;
```

34.Mostre todos os alunos, inclusive os que não estão inscritos em nenhum curso.

```
select alunos.id, alunos.nome, cursos.idcurso as id_cursos, cursos.Nome as nome_cursos
FROM alunos
left JOIN assistem
ON alunos.id = assistem.idalunos
left joinl cursos
ON cursos.idcurso = assistem.idcursos;
```

Exercício sql2

JOIN e INNER JOIN:

1. Liste os alunos e os cursos que estão assistindo.

```
select alunos.Nome, cursos.Nome
from alunos
join assistem
on alunos.id = assistem.idalunos
```

```
 join cursos
on cursos.idcurso = assistem.idcursos
order by alunos.Nome;
```

2. Mostre os alunos e os cursos que estão assistindo usando INNER JOIN.

```
select alunos.Nome, cursos.Nome
from alunos
inner join assistem
on alunos.id = assistem.idalunos
inner join cursos
on cursos.idcurso = assistem.idcursos
order by alunos.Nome;
```

3. Encontre os cursos que não têm alunos inscritos.

```
select cursos.idcurso,cursos.Nome
from cursos
left join assistem
on cursos.idcurso = assistem.idcursos
WHERE assistem.idcursos IS NULL;
```

4. Mostre os alunos que estão inscritos em pelo menos um curso.

```
 select alunos.Nome, cursos.Nome
from alunos
inner join assistem
on alunos.id = assistem.idalunos
inner join cursos
on cursos.idcurso = assistem.idcursos
order by alunos.Nome;
```

5. Liste os alunos que estão assistindo ao curso "php" usando INNER JOIN.

```
select alunos.Nome, cursos.Nome
```

```
from alunos
inner join assistem
on alunos.id = assistem.idalunos
inner join cursos
on cursos.idcurso = assistem.idcursos
where cursos.Nome ='PHP';
```

LEFT JOIN:

6. Liste todos os cursos e, se houver, os alunos inscritos em cada um.

```
SELECT cursos.Nome AS Curso,
GROUP_CONCAT(alunos.Nome SEPARATOR ', ') AS
Alunos_Inscritos
FROM cursos
LEFT JOIN assistem
ON cursos.idcurso = assistem.idcursos
LEFT JOIN alunos
ON assistem.idalunos = alunos.id
GROUP BY cursos.idcurso;
```

7. Mostre os alunos que não estão inscritos em nenhum curso usando LEFT JOIN.

```
select alunos.id, alunos.Nome
from alunos
left join assistem
on alunos.id = assistem.idalunos
where assistem.idalunos is null;
```

8. Encontre os cursos e seus alunos, mesmo que não haja alunos inscritos.

```
SELECT alunos.Nome, cursos.Nome
FROM cursos
LEFT JOIN assistem
ON cursos.idcurso = assistem.idcursos
LEFT JOIN alunos
ON alunos.id = assistem.idalunos;
```

UPDATE:

9. Atualize a idade do aluno com ID 5 para 25 anos.

```
Update alunos
set idade =25
where id =5;
```

10.Incremente em 1 a idade de todos os alunos.

```
update alunos
set idade = idade +1
```

SELECT:

11. Selecione o nome dos alunos em ordem alfabética.

```
Select nome
from alunos
order by nome
```

12. Conte quantos alunos estão inscritos em cada curso.

```
SELECT cursos.idcurso, cursos.nome, COUNT(alunos.id) AS quantidade
FROM cursos
JOIN assistem ON cursos.idcurso = assistem.idcursos
JOIN alunos ON alunos.id = assistem.idalunos
GROUP BY cursos.idcurso, cursos.nome
```

13. Liste os alunos que estão assistindo ao curso com o ID 10 ou ao curso com o ID 20.

```
select alunos.Nome, cursos.Nome
        from alunos
        inner join assistem
        on alunos.id = assistem.idalunos
        inner join cursos
        on cursos.idcurso = assistem.idcursos
        where cursos.idcursos = 10 or cursos.idcursos = 20
```

14. Mostre os cursos que têm a palavra "Avançado" em seus nomes.

```
select nome
from cursos
where nome like ‘%Avançado’;
```

15. Encontre os alunos que estão inscritos em mais de um curso.
Mistura de JOIN e UPDATE:

```
UPDATE alunos
SET campo_de_interesse = 'Mais de um curso'
WHERE id IN (
   SELECT idalunos
   FROM assistem
   GROUP BY idalunos
   HAVING COUNT(DISTINCT idcursos) > 1
);
```

16. Atualize o nome do aluno que está inscrito no curso com o ID 15 para "João Silva".

```
update alunos
set Nome ='João Silva'
where id =35;
```

Mais JOIN e INNER JOIN:

17. Liste os cursos que têm mais alunos inscritos do que o curso com o ID 3.

```
SELECT cursos.idcurso, cursos.nome, COUNT(assistem.idalunos) AS
quantidade_alunos
FROM cursos
INNER JOIN assistem
ON cursos.idcurso = assistem.idcursos
WHERE cursos.idcurso != 3
GROUP BY cursos.idcurso, cursos.nome
HAVING quantidade_alunos > (
   SELECT COUNT(idalunos)
   FROM assistem
   WHERE idcursos = 3)
```

18. Mostre os cursos que têm o mesmo número de alunos inscritos que o curso com o ID 7.

```
SELECT cursos.idcurso, cursos.nome, COUNT(assistem.idalunos) AS
quantidade_alunos
FROM cursos
INNER JOIN assistem
ON cursos.idcurso = assistem.idcursos
WHERE cursos.idcurso = 7
GROUP BY cursos.idcurso, cursos.nome
HAVING quantidade_alunos = (
   SELECT COUNT(idalunos)
   FROM assistem
   WHERE idcursos = 7)
```

LEFT JOIN e UPDATE:

19. Atualize a data de inscrição do aluno com ID 8 para a data atual.

```
UPDATE alunos
SET inscricao = CURDATE()
```

```
WHERE id = 8;
```

SELECT com COUNT e JOIN:

20. Liste os cursos e o número de alunos inscritos em cada um.

```
select cursos.nome, count( alunos.id) as c
from cursos
join assistem
on cursos.idcursos = assistem.idcursos
join alunos
on alunos.id = assistem.idalunos
GROUP BY cursos.nome
ORDER BY cursos.nome, c;
```

Mistura de JOIN e SELECT:

21. Encontre os alunos que estão assistindo ao curso "Inglês Avançado" ou "Matemática Básica".

```
SELECT alunos.Nome, cursos.Nome
FROM alunos
INNER JOIN assistem
ON alunos.id = assistem.idalunos
INNER JOIN cursos
ON cursos.idcurso = assistem.idcursos
WHERE cursos.nome = "Inglês Avançado" OR cursos.nome
= "Matemática Básica";
```

SELECT e LEFT JOIN:

22. Liste os alunos e seus cursos, inclusive aqueles que não estão inscritos em nenhum curso.

```
select alunos.Nome, cursos.Nome
from alunos
```

```
left join assistem
on alunos.id = assistem.idcursos
left join cursos
on cursos.idcurso = assistem.idcursos
```

UPDATE com JOIN:

23. Atualize o nome do curso com ID 25 para "Espanhol Intermediário".

```
update cursos
set nome=  'Espanhol Intermediário'
WHERE idcursos = 25;
```

SELECT com ORDER BY e JOIN:

24. Liste os cinco primeiros alunos em ordem alfabética.

```
select Nome
from alunos
order by nome
limit 5
```

SELECT com DISTINCT(seja, não haja duplicatas) e JOIN:

25. Liste os nomes únicos de todos os cursos que têm alunos inscritos.

```
SELECT DISTINCT cursos.nome
FROM cursos
JOIN assistem
ON cursos.idcurso = assistem.idcursos;
```

LEFT JOIN e SELECT com SUM:

26. Liste os alunos e a soma das idades de todos os alunos inscritos para cada um.

```
SELECT Nome, SUM(idade)
```

```
FROM alunos
GROUP BY Nome;
```

Mistura de JOIN e SELECT com WHERE:

27. Liste os alunos que têm menos de 20 anos e estão inscritos no curso "Química".

```
select alunos.Nome, cursos.Nome
from alunos
inner join assistem
on alunos.id = assistem.idalunos
inner join cursos
on cursos.idcurso = assistem.idcursos
where alunos.idade<20 and cursos.Nome = ‘Química’;
```

SELECT com GROUP BY e JOIN:

28. Liste os cursos e a idade média dos alunos inscritos em cada um.

```
select cursos.Nome,  AVG(alunos.idade) as media
from alunos
join assistem
on alunos.id = assistem.idalunos
join cursos
on cursos.idcurso = assistem.idcursos
GROUP BY cursos.Nome;
```

Mais SELECT com JOIN:

29. Encontre os cursos e os alunos inscritos que têm mais de 25 anos.

```
select alunos.Nome, cursos.Nome
from alunos
inner join assistem
```

```
on alunos.id = assistem.idalunos
inner join cursos
on cursos.idcurso = assistem.idcursos
where alunos.idade>25;
```

Mistura de UPDATE e JOIN:

30. Atualize a data de inscrição de todos os alunos que estão inscritos no curso "Programação Avançada" para a data atual.

```
UPDATE alunos
SET inscricao = CURDATE()
WHERE id IN (
SELECT idalunos
FROM assistem
WHERE idcursos
IN ( SELECT idcurso
FROM cursos
WHERE nome = "Programação Avançada" ));
```

OBSERVAÇÕES

WHERE | HAVING

WHERE: A cláusula `WHERE` é usada em consultas para filtrar registros antes que eles sejam agrupados ou retornados. Ela é usada para aplicar condições aos registros individuais antes de qualquer agregação.

HAVING: A cláusula `HAVING` é usada em conjunto com `GROUP BY` para filtrar os resultados após a agregação. Ela permite que você aplique condições a grupos de registros após a agregação ter sido realizada

ORDER BY | GROUP BY

ORDER BY: A cláusula `ORDER BY` é usada para ordenar os resultados da consulta de acordo com uma ou mais colunas. Ela é usada para especificar a ordem em que as linhas serão exibidas no resultado.

GROUP BY: A cláusula `GROUP BY` é usada para agrupar os resultados da consulta em grupos com base nos valores de uma ou mais colunas. Ela é frequentemente usada em conjunto com funções de agregação, como `COUNT`, `SUM`, `AVG`, etc. O resultado é uma agregação dos dados para cada grupo identificado pelas colunas especificadas no `GROUP BY`.

JOIN | LEFT JOIN

***OIN*: A palavra-chave `JOIN` é usada para combinar dados de duas ou mais tabelas. No entanto, quando você usa apenas `JOIN`, ele é tratado como um `INNER JOIN` por padrão. Isso significa que somente os registros que têm correspondências em ambas as tabelas serão incluídos no resultado**

LEFT JOIN: A cláusula `LEFT JOIN` é usada para combinar dados entre tabelas, incluindo todos os registros da tabela à esquerda (primeira tabela) e apenas os registros correspondentes da tabela à direita (segunda tabela). Se não houver correspondência, as colunas da tabela à direita serão preenchidas com valores NULL.