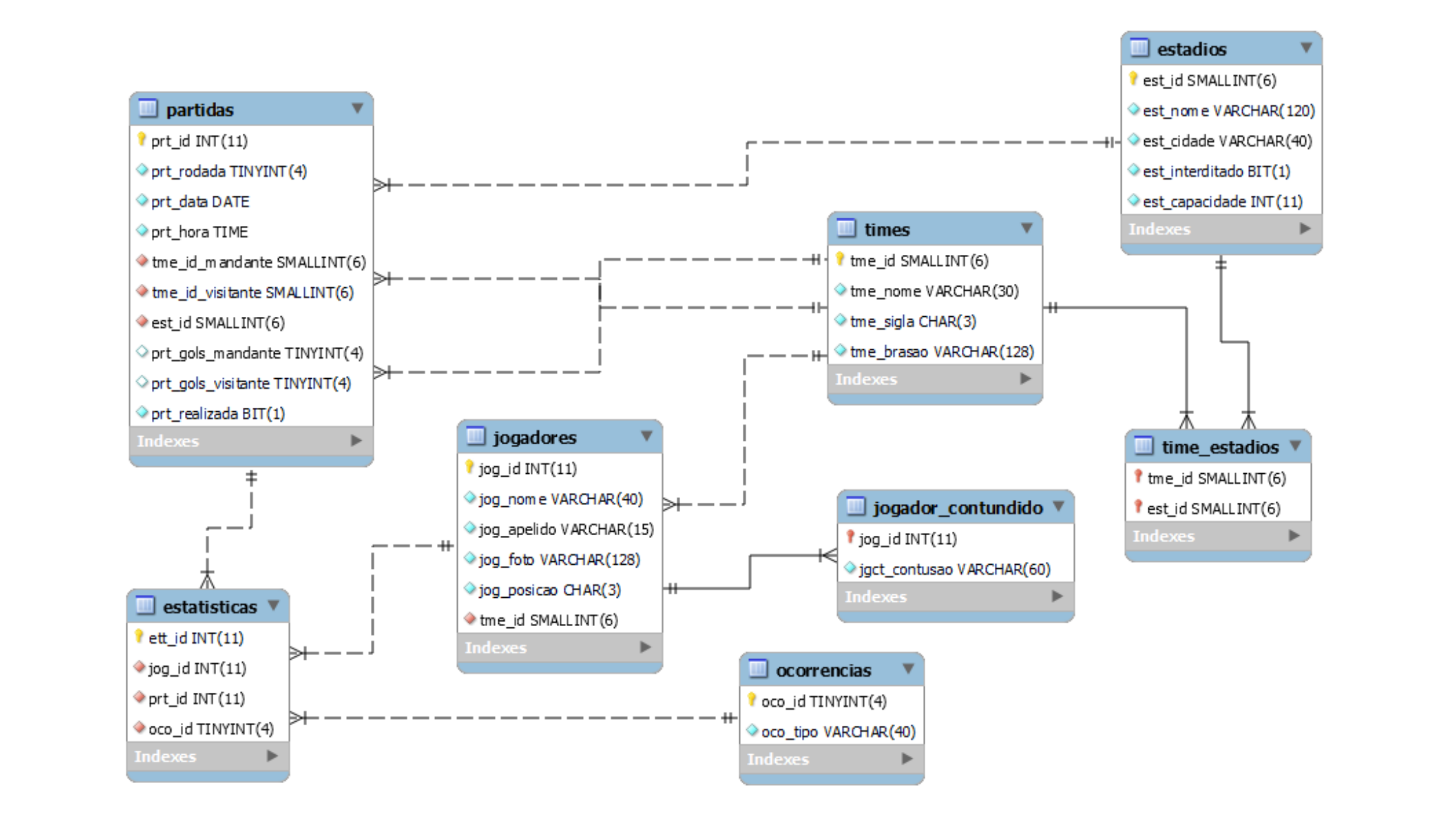
JOINS

Prof. Ewerton

JOINS

O comando JOIN é utilizado para combinar registros de duas ou mais tabelas com base em uma condição especificada. Ele permite que você recupere dados relacionados de tabelas diferentes em uma única consulta.

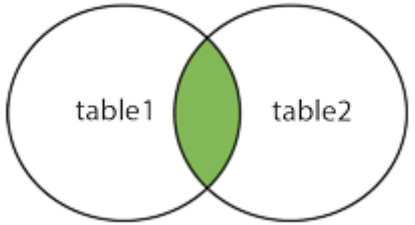
Existem diferentes tipos de JOIN disponíveis no MySQL:


partidas
prt_id INT(11)
prt_rodada TINYINT(4)
prt_data DATE
prt_hora TIME
tme_id_mandante SMALLINT(6)
tme_id_visitante SMALLINT(6)
est_id SMALLINT(6)
prt_gols_mandante TINYINT(4)
prt_gols_visitante TINYINT(4)
prt_realizada BIT(1)
Indexes
estadios
est_id SMALLINT(6)
est_nome VARCHAR(120)
est_cidade VARCHAR(40)
est_interditado BIT(1)
est_capacidade INT(11)
Indexes
times
tme_id SMALLINT(6)
tme_nome VARCHAR(30)
tme_sigla CHAR(3)
tme_brasao VARCHAR(128)
Indexes
time_estadios
tme_id SMALLINT(6)
est_id SMALLINT(6)
Indexes
jogadores
jog_id INT(11)
jog_nome VARCHAR(40)
jog_apelido VARCHAR(15)
jog_foto VARCHAR(128)
jog_posicao CHAR(3)
tme_id SMALLINT(6)
Indexes
jogador_contundido
jog_id INT(11)
jgct_contusao VARCHAR(60)
Indexes
estatisticas
ett_id INT(11)
jog_id INT(11)
prt_id INT(11)
oco_id TINYINT(4)
Indexes
ocorrencias
oco_id TINYINT(4)
oco_tipo VARCHAR(40)
Indexes

INNER JOIN

O INNER JOIN retorna apenas os registros que possuem correspondência em ambas as tabelas. Ele combina os registros com base na condição especificada após a cláusula ON. Ex:

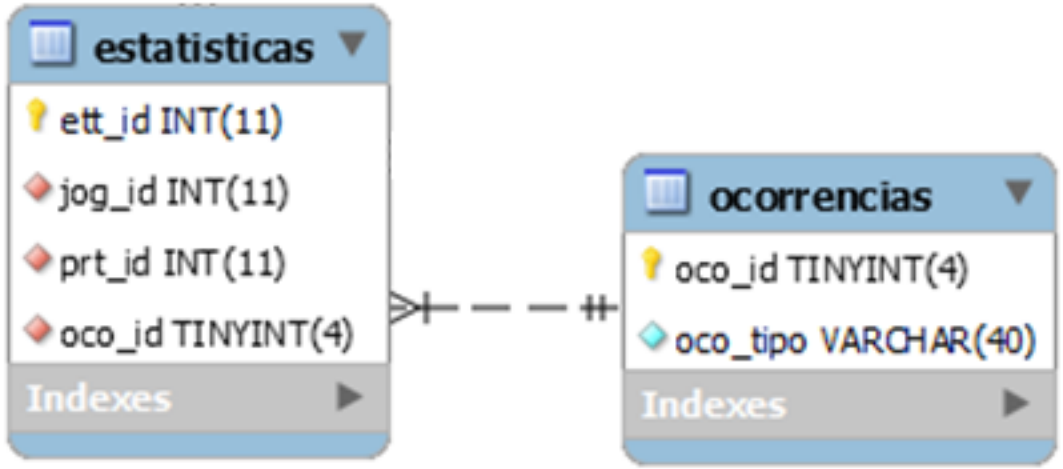
```
SELECT coluna1, coluna2, ...
FROM tabela1
INNER JOIN tabela2 ON tabela1.coluna = tabela2.coluna;
```



Mostrar o nome das ocorrências na partida

```
SELECT e.ett_id, e.jog_id, e.prt_id, o.oco_tipo
FROM estatisticas e
INNER JOIN ocorrencias o ON e.oco_id = o.oco_id;
```

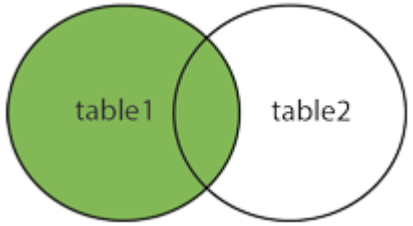
ett_id	jog_id	prt_id	oco_tipo
1	39	1	Gol
2	39	1	Gol
3	46	1	Cartão Amarelo
4	147	3	Gol
5	129	3	Gol
6	52	3	Cartão Amarelo
7	61	3	Cartão Amarelo
8	51	3	Gol
9	52	3	Gol
10	39	4	Gol
11	31	4	Cartão Amarelo
12	93	4	Cartão Amarelo



LEFT JOIN

O LEFT JOIN retorna todos os registros da tabela da esquerda (tabela1) e os registros correspondentes da tabela da direita (tabela2). Se não houver correspondência, ele retorna NULL para as colunas da tabela da direita. Ex:

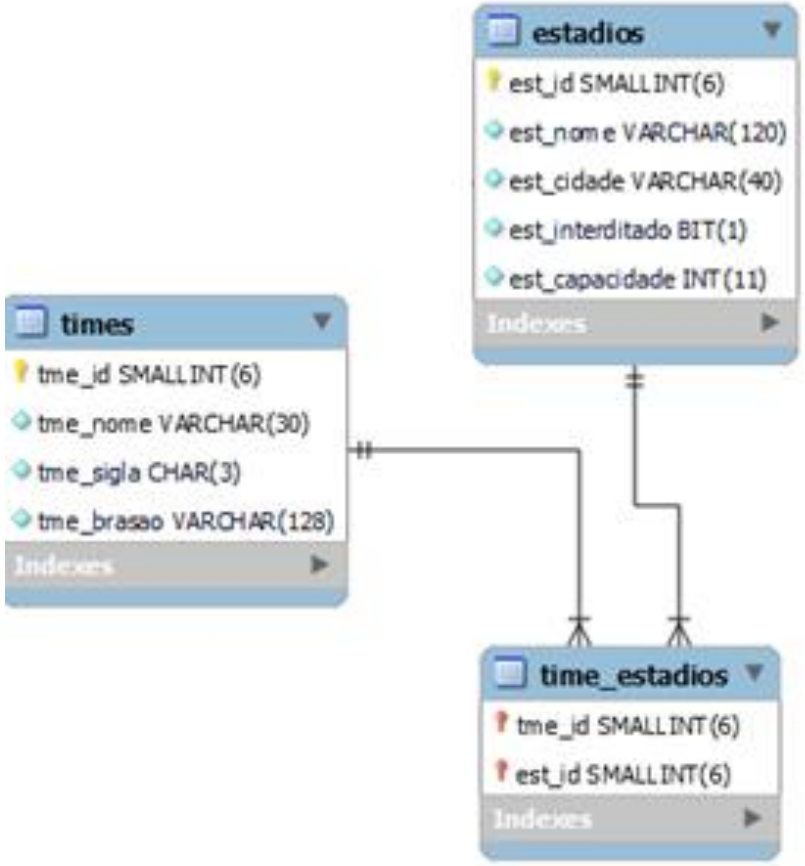
```
SELECT coluna1, coluna2, ...
FROM tabela1
LEFT JOIN tabela2 ON tabela1.coluna = tabela2.coluna;
```



LISTA TODOS OS ESTÁDIOS, MESMO QUE NÃO EXISTA UM TIME ASSOCIADO A ELE

```sql
SELECT e.est_id, e.est_nome, e.est_cidade, e.est_interditado,
       e.est_capacidade, te.tme_id
FROM estadios e
LEFT JOIN time_estadios te ON e.est_id = te.est_id;
```

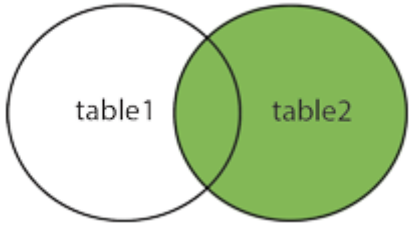
est_id	est_nome	est_cidade	est_interditado	est_capacidade	tme_id
1	Neo Química Arena	São Paulo	0	47605	1
2	Allianz Parque	São Paulo	0	43713	2
3	Vila Belmiro	Santos	0	17923	3
4	Estádio Municipal José Maria de Campos Maia	Mirassol	0	19000	5
5	Estádio Moisés Lucarelli	Campinas	0	17728	6
6	Pacaembu	São Paulo	1	0	NULL
7	Arena Barueri	Barueri	0	31452	2
8	Canindé	São Paulo	0	21004	7



RIGHT JOIN

O RIGHT JOIN é o oposto do LEFT JOIN. Ele retorna todos os registros da tabela da direita (tabela2) e os registros correspondentes da tabela da esquerda (tabela1). Se não houver correspondência, ele retorna NULL para as colunas da tabela da esquerda. Ex:

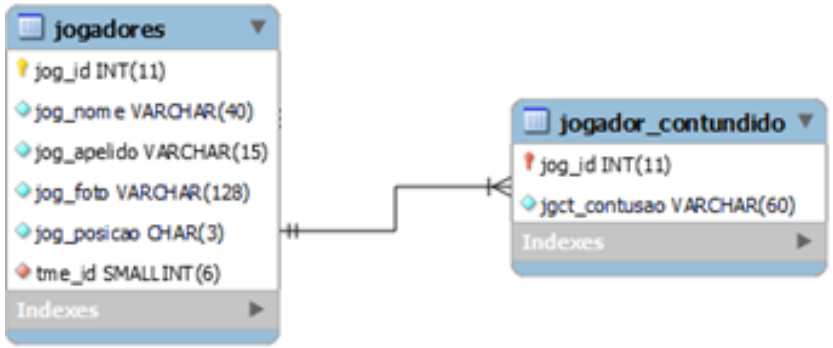
```
SELECT coluna1, coluna2, ...
FROM tabela1
RIGHT JOIN tabela2 ON tabela1.coluna = tabela2.coluna;
```



LISTAR APENAS JOGADORES CONTUNDIDOS DE DETERMINADO TIME

```
SELECT j.jog_nome, j.jog_posicao, jgct_contusao FROM jogadores j
RIGHT JOIN jogador_contundido jc ON j.jog_id = jc.jog_id
WHERE j.tme_id = 2;
```

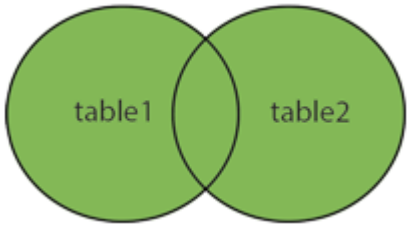
jog_nome	jog_posicao	jgct_contusao
Thiago Gonçalves	GOL	Quebrou o braço
Lorenzo Gonçalves	GOL	Perdeu o dedo
Carlos Eduardo Nogueira	ATA	Torceu o joelho



FULL JOIN:

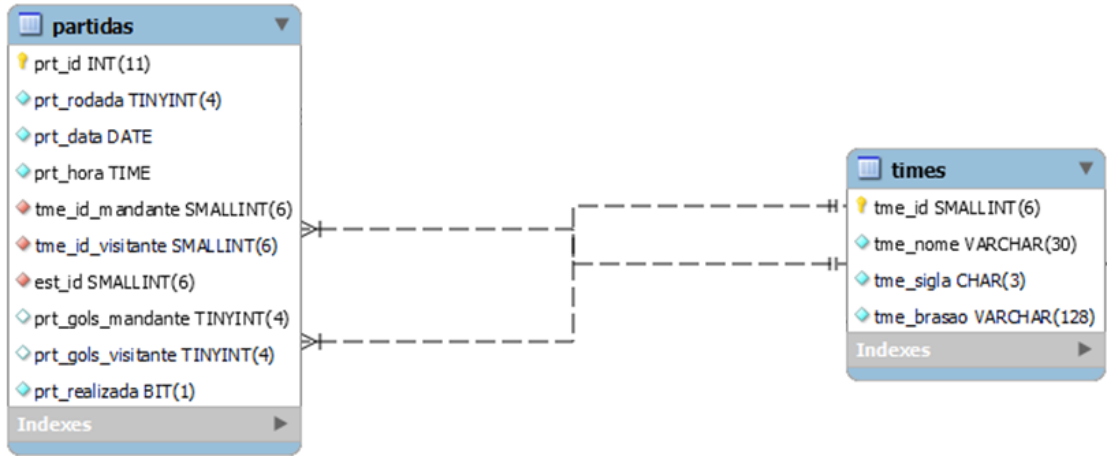
O FULL JOIN retorna todos os registros de ambas as tabelas, combinando-os com base na condição especificada. Se não houver correspondência, ele retorna NULL para as colunas não correspondentes. Ex:

```
SELECT coluna1, coluna2, ...
FROM tabela1
FULL JOIN tabela2 ON tabela1.coluna = tabela2.coluna;
```



No MySQL, não existe um comando FULL JOIN nativo. No entanto, é possível obter um resultado semelhante usando um LEFT JOIN e um RIGHT JOIN combinados. Aqui estão alguns exemplos de como realizar um FULL JOIN com base na estrutura fornecida:

```sql
SELECT t.tme_id, t.tme_nome, t.tme_sigla, t.tme_brasao,
       p.prt_id, p.prt_rodada, p.prt_data, p.prt_hora,
       p.est_id, p.tme_id_mandante, p.prt_gols_mandante,
       p.tme_id_visitante, p.prt_gols_visitante, p.prt_realizada
FROM times t
LEFT JOIN partidas p ON t.tme_id = p.tme_id_mandante
UNION
SELECT t.tme_id, t.tme_nome, t.tme_sigla, t.tme_brasao,
       p.prt_id, p.prt_rodada, p.prt_data, p.prt_hora,
       p.est_id, p.tme_id_mandante, p.prt_gols_mandante,
       p.tme_id_visitante, p.prt_gols_visitante, p.prt_realizada
FROM times t
RIGHT JOIN partidas p ON t.tme_id = p.tme_id_mandante;
```



tme_id	tme_nome	tme_sigla	tme_brasao	prt_id	prt_rodada	prt_data	prt_hora	est_id	tme_id_mandante	prt_gols_mandante	tme_id_visitante	prt_gols_visitante	prt_realizada
2	Palmeiras	PAL	palmeiras.png	5	2	2023-03-05	19:00:00	2	2	0	6	3	1
2	Palmeiras	PAL	palmeiras.png	12	4	2023-03-13	19:00:00	7	2	1	5	2	1
2	Palmeiras	PAL	palmeiras.png	15	5	2023-03-17	20:00:00	2	2	NULL	3	NULL	0
3	Santos	SAN	santos.png	2	1	2023-03-01	16:00:00	3	3	0	5	0	1
3	Santos	SAN	santos.png	4	2	2023-03-04	20:00:00	3	3	0	1	1	1
3	Santos	SAN	santos.png	11	4	2023-03-12	22:00:00	3	3	0	7	0	1
4	São Paulo	SAO	saopaulo.png	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
5	Mirassol	MIR	mirassol.png	6	2	2023-03-05	21:00:00	5	5	2	7	1	1
5	Mirassol	MIR	mirassol.png	14	5	2023-03-16	19:00:00	5	5	NULL	6	NULL	0
6	Ponte Preta	PTA	ponte.png	3	1	2023-03-01	19:00:00	6	6	2	7	2	1
6	Ponte Preta	PTA	ponte.png	9	3	2023-03-09	18:00:00	6	6	0	3	0	1

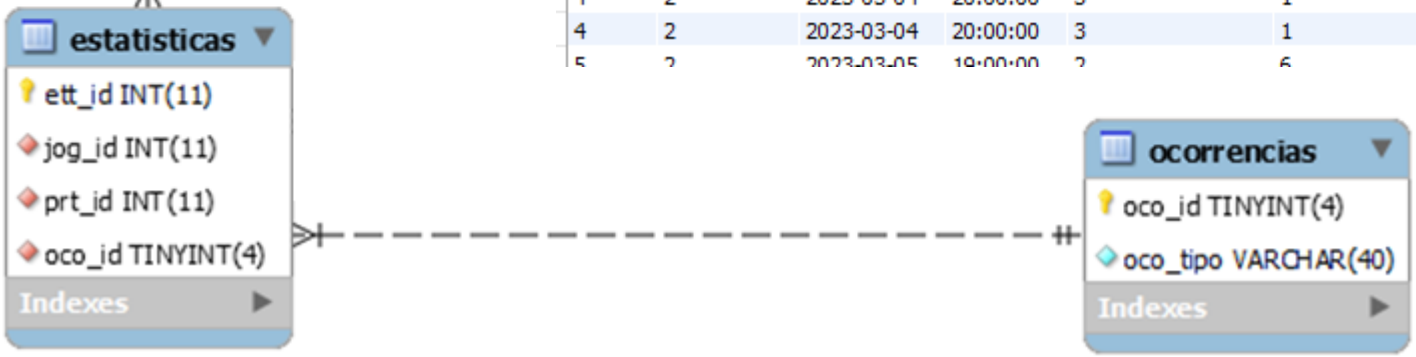
CROSS JOIN

O CROSS JOIN é usado para obter o produto cartesiano entre duas tabelas, ou seja, ele combina cada linha de uma tabela com cada linha de outra tabela. Como resultado, o número total de linhas retornadas será o produto do número de linhas nas tabelas envolvidas. Ex:

```
SELECT *
FROM tabela1
CROSS JOIN tabela2;
```

O exemplo acima irá combinar todas as linhas da tabela1 com todas as linhas da tabela2, retornando todas as combinações possíveis.

FORÇAR UMA RELAÇÃO ENTRE PARTIDAS E OCORRÊNCIAS




```
SELECT *
FROM partidas p
CROSS JOIN ocorrencias o;
```

prt_id	prt_rodada	prt_data	prt_hora	tme_id_mandante	tme_id_visitante	est_id	prt_gols_mandante	prt_gols_visitante	prt_realizada	oco_id	oco_tipo
1	1	2023-03-01	16:00:00	1	2	1	2	0	1	1	Gol
1	1	2023-03-01	16:00:00	1	2	1	2	0	1	2	Cartão Amarelo
1	1	2023-03-01	16:00:00	1	2	1	2	0	1	3	Cartão Vermelho
2	1	2023-03-01	16:00:00	3	5	3	0	0	1	1	Gol
2	1	2023-03-01	16:00:00	3	5	3	0	0	1	2	Cartão Amarelo
2	1	2023-03-01	16:00:00	3	5	3	0	0	1	3	Cartão Vermelho
3	1	2023-03-01	19:00:00	6	7	6	2	2	1	1	Gol
3	1	2023-03-01	19:00:00	6	7	6	2	2	1	2	Cartão Amarelo
3	1	2023-03-01	19:00:00	6	7	6	2	2	1	3	Cartão Vermelho
4	2	2023-03-04	20:00:00	3	1	3	0	1	1	1	Gol
4	2	2023-03-04	20:00:00	3	1	3	0	1	1	2	Cartão Amarelo
4	2	2023-03-04	20:00:00	3	1	3	0	1	1	3	Cartão Vermelho
5	2	2023-03-05	19:00:00	2	6	2	0	3	1	1	Gol

SELF JOIN

O SELF JOIN, como o nome sugere, é usado para combinar uma tabela consigo mesma. Ele é útil quando você precisa comparar registros dentro da mesma tabela com base em uma condição. Ex:

```
SELECT t1.coluna, t2.coluna
FROM tabela t1
INNER JOIN tabela t2 ON t1.coluna = t2.coluna;
```

Nesse exemplo, a tabela é renomeada como t1 e t2, e em seguida é feito um INNER JOIN entre os dois aliases da tabela com base na condição especificada. Isso permite comparar registros da mesma tabela.

O SELF JOIN pode ser útil para consultar hierarquias, como tabelas que possuem relacionamentos de pai-filho, ou para comparar registros com base em valores semelhantes.

```sql
-- Encontrar estádios localizados na mesma cidade
SELECT e1.est_nome, e2.est_nome, e1.est_cidade
FROM estadios e1
INNER JOIN estadios e2 ON e1.est_cidade = e2.est_cidade
WHERE e1.est_id <> e2.est_id;
```

est_nome	est_nome	est_cidade
Allianz Parque	Neo Química Arena	São Paulo
Pacaembu	Neo Química Arena	São Paulo
Canindé	Neo Química Arena	São Paulo
Neo Química Arena	Allianz Parque	São Paulo
Pacaembu	Allianz Parque	São Paulo
Canindé	Allianz Parque	São Paulo
Neo Química Arena	Pacaembu	São Paulo
Allianz Parque	Pacaembu	São Paulo
Canindé	Pacaembu	São Paulo
Neo Química Arena	Canindé	São Paulo
Allianz Parque	Canindé	São Paulo
Pacaembu	Canindé	São Paulo

Lembre-se de que, ao usar o CROSS JOIN ou o SELF JOIN, é importante ter cuidado, pois podem resultar em grandes conjuntos de resultados e consultas pesadas. Certifique-se de utilizar esses tipos de junção com cuidado e de acordo com a necessidade específica do seu cenário.