

Sprint 5 - entrega

Treinamento de Aulas Particulares

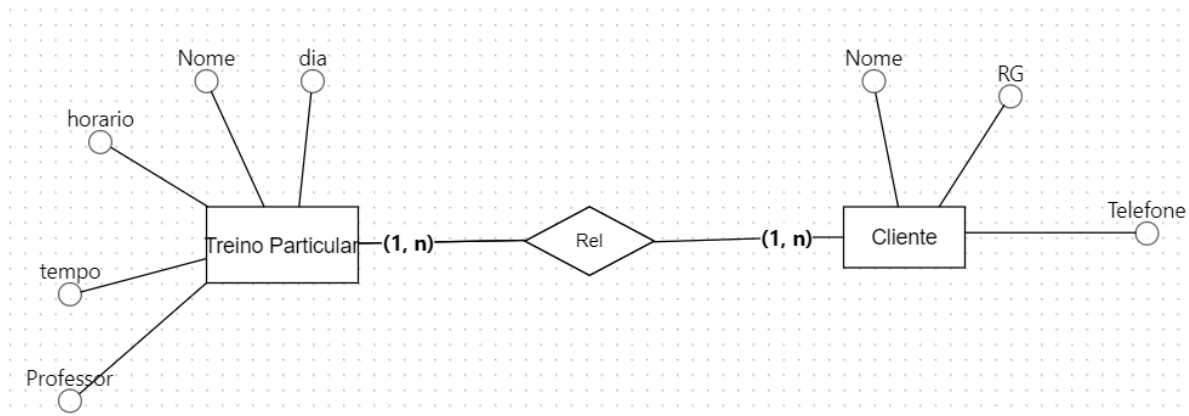
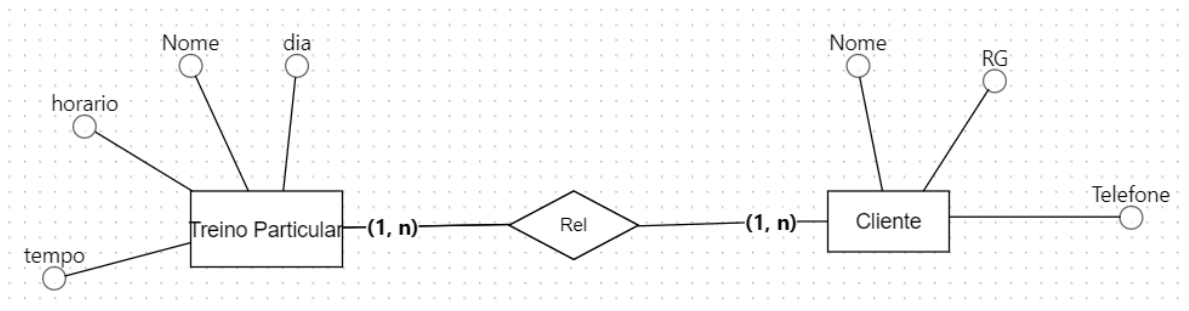
O cliente terá a oportunidade de fazer o seu treino particular na nossa academia.

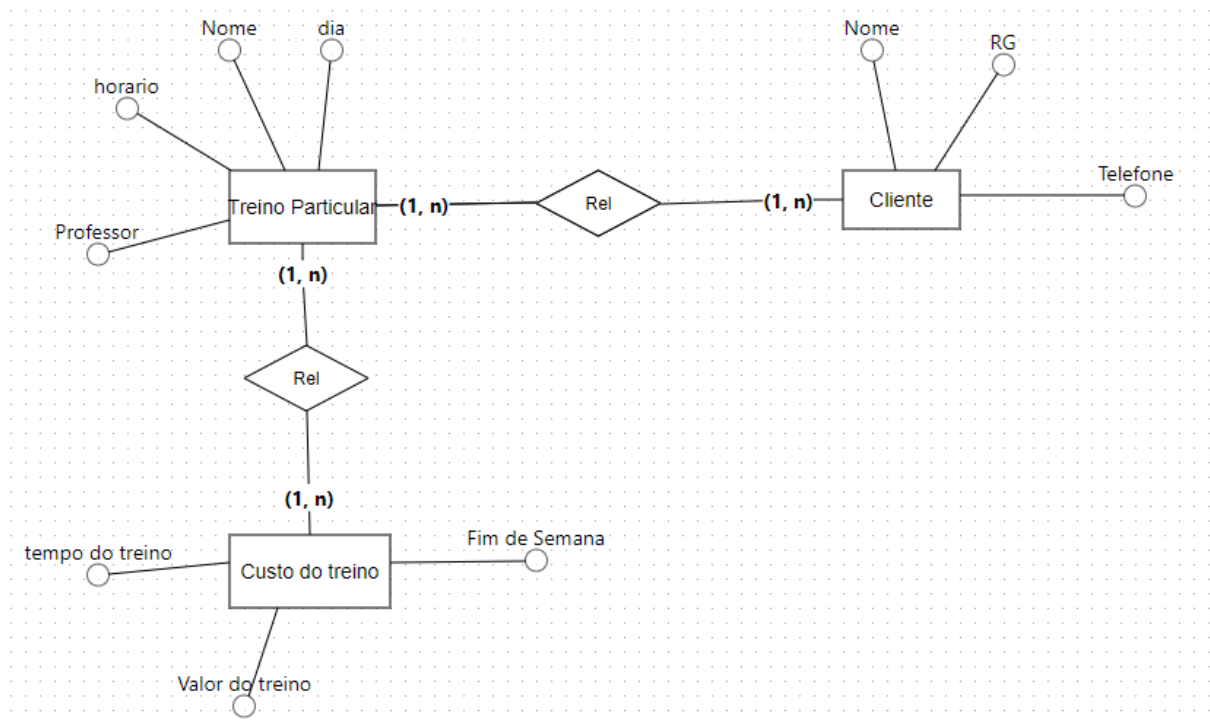
O treino particular precisa ser agendado no dia e horário, e tem a possibilidade de escolher o professor (caso esteja disponível).

O custo do treino particular depende do tempo de duração do treino, sendo os valores colocados na tabela, os valores padrões para o tempo indicado.

Havendo a escolha ou necessidade de ter o treino particular no fim de semana, logo será acrescentado uma taxa de 20% .

1 - Modelagem conceitual





Necessário otimizar o banco de dados, através da normalização 3FN

Treino Nome	Dia - Data	Horário	Professor	código cliente	Cliente	Contato - telefone	RG	Valor do treino	Tempo do treino (minutos)	Treino - Fim de Semana (Acrescimo de 20%)
Cross	01/01/24	05h00	Alice	01	Paulo	06454654564	353536767	100	60	Sim
Musculação	05/02/24	07h00	José	02	Tiago	05184849849	8787653	100	90	não
Fitness	11/03/24	08h00	João	03	Tadeu	0215674892	535557	50	50	não
Yoga	12/04/24	10h00	Allex	04	Ronaldo	02156579448	33646	150	40	Sim
Pilatos	04/05/24	11h00	Gabi	05	Marcelo	02195555555	5353535	200	120	não

3FN

Tabela Cliente

ID_pk código de cliente	Cliente	Contato	RG
01	Paulo	06454654564	353536767
02	Tiago	05184849849	8787653-SP
03	Tadeu	0215674892	535557-RJ
04	Ronaldo	(021)56579448	33646
05	Marcelo	02195555555	5353535-SP

Tabela Custo

ID_pk => Tempo do treino (minuto)	Valor do treino (\$R Real)	Treino - Fim de Semana (Acrescimo de 20%)
60	100	Sim
90	100	não
50	50	não
40	150	Sim
120	200	não

Tabela Treino

ID_pk	Treino Nome	Dia - Data	Horário	Professor
01	Cross	01/01/24	05h00	Alice
02	Musculação	05/02/24	07h00	José
03	Fitness	11/03/24	08h00	João
04	Yoga	12/04/24	10h00	Allex
05	Pilatos	04/05/24	11h00	Gabi

Tabela Criada Exercicio

Relacionamento da **Tabela Treino** com a **tabela Cliente**

Tabela EXERCICIO

fk_id_cliente	fk_id_treino	id_PK
01	01	1
02	02	2
03	03	3
04	04	4
05	05	5

/* Criação no SQL */

/* criar a tabela CUSTO com os seus atributos */

```
CREATE TABLE custo (  
    tempo INT NOT NULL PRIMARY KEY,  
    valor_treino FLOAT NOT NULL,  
    fimSemana BOOLEAN NOT NULL /* 1=> Fim de semana , 0=> durante a semana */  
);
```

/* criar a tabela CLIENTE com os seus atributos */

```
CREATE TABLE cliente (  
    id SERIAL PRIMARY KEY,  
    nome VARCHAR(48) NOT NULL,  
    rg VARCHAR(10) NOT NULL,  
    telefone VARCHAR (12) NOT NULL  
);
```

/* criar a tabela Treino com os seus atributos */

```
CREATE TABLE treino (  
    /*id SERIAL PRIMARY KEY,*/  
    treino VARCHAR(48) NOT NULL,  
    dia VARCHAR(10) NOT NULL,  
    horario VARCHAR (12) NOT NULL,  
    professor VARCHAR (12) NOT NULL  
);
```

/*Agora vamos inserir valores nas nossas tabelas */

/* Inserir Valores na nossas tabelas TREINO , CLIENTES e CUSTO*/

/*inserir Valores tabela do CLIENTE*/

```
INSERT INTO cliente (nome, rg, telefone) VALUES  
    ('Paulo','353536767', '0645465454'),  
    ('Tiago','8787653-SP', '0518484949'),  
    ('Tadeu','535557-RJ', '021567482'),  
    ('Ronaldo','33646','(021)5657948'),  
    ('Marcelo','5353535-SP','(021)955555')
```

```

31 /*inserir tabela do CLIENTE*/
32 INSERT INTO cliente (nome, rg, telefone) VALUES
33     ('Paulo','353536767', '0645465454'),
34     ('Tiago','8787653-SP', '0518484949'),
35     ('Tadeu','535557-RJ', '021567482'),
36     ('Ronaldo','33646','(021)5657948'),
37     ('Marcelo','5353535-SP','(021)955555')
38
39 SELECT * FROM cliente

```

id	nome	rg	telefone
1	Paulo	353536767	0645465454
2	Tiago	8787653-SP	0518484949
3	Tadeu	535557-RJ	021567482
4	Ronaldo	33646	(021)5657948
5	Marcelo	5353535-SP	(021)955555

/* Inserir dados na tabela TREINO */

```

INSERT INTO treino (treino, dia, horario,professor) VALUES
    ('cross','01/01/24', '05h00','Alice'),
    ('musculacao','05/02/24', '07h00','José'),
    ('fitness','11/03/24', '08h00','João'),
    ('yoga','12/04/24','10h00','Alex'),
    ('Pilatos','04/05/24','11h00','Gabi')

```

```

50
51 /* Inserir dados na tabela TREINO */
52 INSERT INTO treino (treino, dia, horario, professor) VALUES
53     ('cross', '01/01/24', '05h00', 'Alice'),
54     ('musculacao', '05/02/24', '07h00', 'José'),
55     ('fitness', '11/03/24', '08h00', 'João'),
56     ('yoga', '12/04/24', '10h00', 'Alex'),
57     ('Pilatos', '04/05/24', '11h00', 'Gabi')
58
59 SELECT * FROM treino
60
61

```

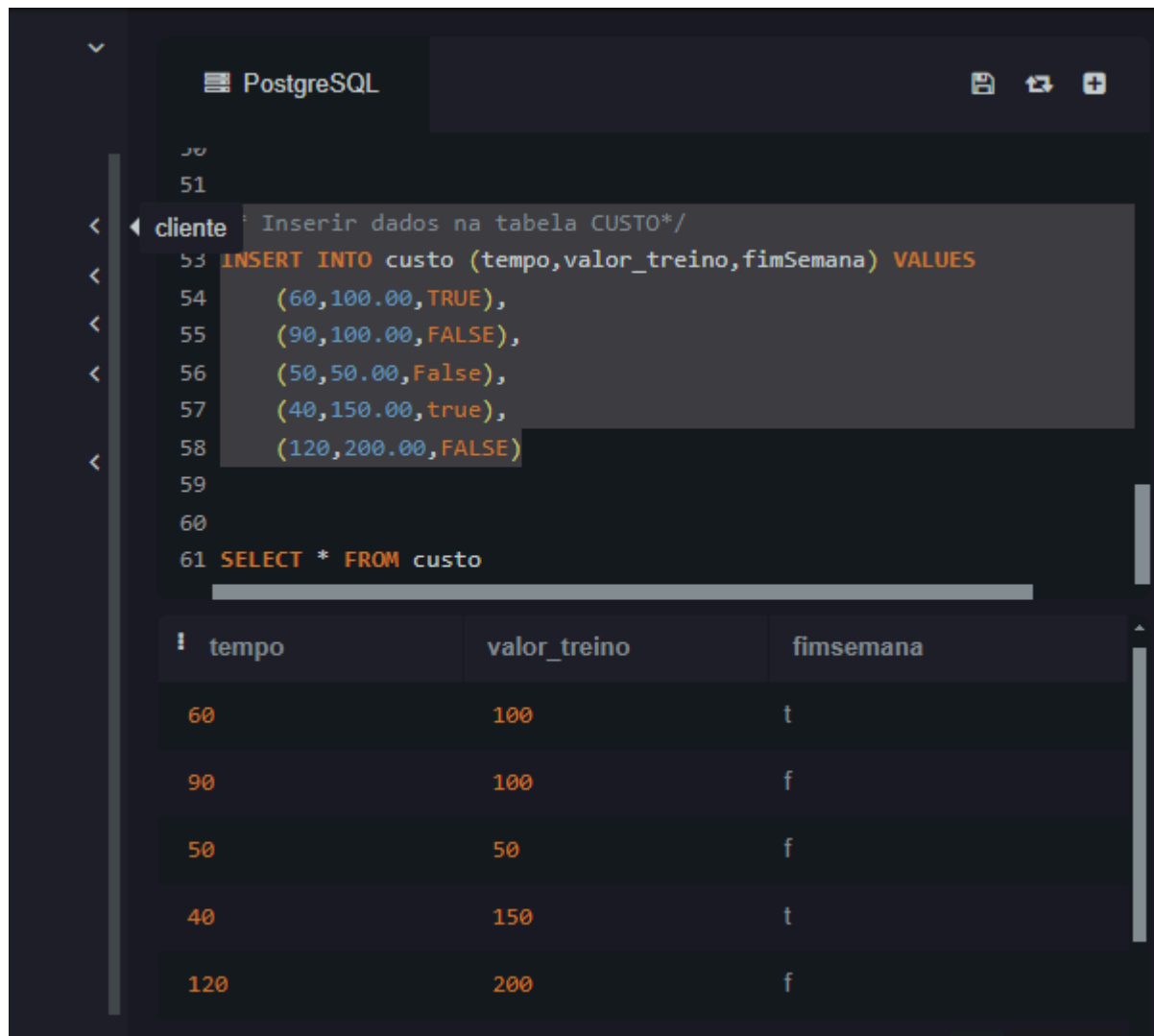
!	id	treino	dia	horario	professor
1		cross	01/01/24	05h00	Alice
2		musculacao	05/02/24	07h00	José
3		fitness	11/03/24	08h00	João
4		yoga	12/04/24	10h00	Alex
5		Pilatos	04/05/24	11h00	Gabi

/* Inserir dados na tabela CUSTO*/

```

INSERT INTO custo (tempo, valor_treino, fimSemana) VALUES
    (60, 100.00, TRUE),
    (90, 100.00, FALSE),
    (50, 50.00, False),
    (40, 150.00, true),
    (120, 200.00, FALSE)

```



/* Agora como funcionaria a nossa tabela auxiliar */

/* Associar o TREINO com CLIENTE */

/* Criar a tabela EXERCICIO*/

```
CREATE TABLE exercicio (  
    id SERIAL PRIMARY KEY,  
    fk_id_treino INT NOT NULL,  
    fk_id_cliente INT NOT NULL,  
    CONSTRAINT fk_treino FOREIGN KEY (fk_id_treino) REFERENCES treino(ID),  
    CONSTRAINT fk_cliente FOREIGN KEY (fk_id_cliente) REFERENCES cliente(ID)  
);
```

/* Inserindo valores na tabela auxiliar EXERCICIO */

```
INSERT INTO exercicio (fk_id_treino,fk_id_cliente) VALUES  
(1,1),
```

(2,2),
(3,3),
(4,4),
(5,5)

Table	88
cliente	89 SELECT * FROM exercicio
custo	90
demo	91 /* Inserindo valores na tabela auxiliar EXERCICIO */
exercicio	92 INSERT INTO exercicio (fk_id_treino,fk_id_cliente) VALUES
Column	93 (1,1),
id integer	94 (2,2),
fk_id_treino integer	95 (3,3),
fk_id_cliente integer	96 (4,4),
Trigger	97 (5,5)
RI_ConstraintTrigger...	98
RI_ConstraintTrigger...	99
RI_ConstraintTrigger...	
RI_ConstraintTrigger...	
treino	
MS SQL	

id	fk_id_treino	fk_id_cliente
1	1	1
2	2	2
3	3	3
4	4	4
5	5	5

/* Fazendo relacionamentos Tabela CLIENTE e tabela TREINO*/

/* selecionar uma tabela com os registros da tabela TREINO */

/* e com os registros em comum da tabela CLIENTE */

SELECT * FROM treino /* tabela á esquerda*/

inner join cliente /* pegar todos os registros em comum entre tabelas */

on treino.id = cliente.id /* comando on é tipo indicar qual a conexão entre colunas das tabelas precisam ser iguais */


```

101 Fazendo relacionamentos Tabela CLIENTE e tabela TREINO*/
102 selecionar uma tabela com os registros da tabela TREINO */
103: com os registros em comum da tabela CLIENTE */
104
105.ECT * FROM treino /* tabela á esquerda*/
106 INNER JOIN cliente /* pegar todos os registros em comum entre tabelas */
107 ON treino.id = cliente.id /* comando on é tipo indicar qual a conexão entre
108
109

```

id	treino	dia	hor...	prof...	id	nome	rg	telefone
1	cross	01/0...	05h00	Alice	1	Paulo	353...	06454654...
2	musc...	05/0...	07h00	José	2	Tiago	8787...	05184849...
3	fitness	11/03...	08h00	João	3	Tadeu	5355...	021567482
4	yoga	12/0...	10h00	Alex	4	Rona...	336...	(021)5657...
5	Pilatos	04/0...	11h00	Gabi	5	Marc...	5353...	(021)955555

/* Agora como funcionaria a nossa tabela auxiliar */

/* Associar o TREINO com CUSTO */

/* Criar a tabela ORCAMENTO*/

```

CREATE TABLE orcamento (
    id SERIAL PRIMARY KEY,
    fk_id_treino INT NOT NULL,
    fk_id_custo INT NOT NULL,
    CONSTRAINT fk_treino FOREIGN KEY (fk_id_treino) REFERENCES treino(ID),
    CONSTRAINT fk_custo FOREIGN KEY (fk_id_custo) REFERENCES custo(tempo)
);

```

/* Inserir valores na tabela ORCAMENTO */

```

INSERT INTO orcamento (fk_id_treino,fk_id_custo) VALUES
    (1,60),
    (2,90),
    (3,50),
    (4,40),
    (5,120)

```

```

SELECT * FROM orcamento

```

```

123 /* Inserir valores na tabela ORCAMENTO */
124 INSERT INTO orcamento (fk_id_treino,fk_id_custo) VALUES
125     (1,60),
126     (2,90),
127     (3,50),
128     (4,40),
129     (5,120)
130
131 SELECT * FROM orcamento
132
133
134

```

id	fk_id_treino	fk_id_custo
6	1	60
7	2	90
8	3	50
9	4	40
10	5	120

/*Fazendo relacionamentos Tabela CUSTO e tabela ORCAMENTO */
/* selecionar uma tabela com os registros da tabela ORCAMENTO */
/* e com os registros em comum da tabela CUSTO */

```

SELECT * FROM orcamento /* tabela á esquerda*/
    inner join custo /* pegar todos os registros em comum entre tabelas */
    ON orcamento.fk_id_custo = custo.tempo

```

```

133
134  /*Fazendo relacionamentos Tabela CUSTO e tabela ORCAMENTO */
135  /* selecionar uma tabela com os registros da tabela ORCAMENTO */
136  /* e com os registros em comum da tabela CUSTO */
137
138  SELECT * FROM orcamento /* tabela á esquerda*/
139  INNER JOIN custo /* pegar todos os registros em comum entre tabelas
140  ON orcamento.fk_id_custo = custo.tempo

```

!	id	fk_id_trei...	fk_id_cu...	tempo	valor_tre...	fimsemana
	6	1	60	60	100	t
	7	2	90	90	100	f
	8	3	50	50	50	f
	9	4	40	40	150	t
	10	5	120	120	200	f

Sendo aula no fim-de-semana, deverá ser aplicado adicional de 20%.
 Função para aplicar adicional de 20% na tabela de dados.

/* escrever uma função Aplicar_adicional 20%*/

```

CREATE or REPLACE FUNCTION aplicar_adicional (preco FLOAT, adicional FLOAT)
  RETURNS FLOAT as $$
  BEGIN
    RETURN preco * (1 + adicional);
  END
  $$ LANGUAGE PLPGSQL /*PROCEDURAL LANGUAGE postgresql */

```

```

SELECT aplicar_adicional (10,0.2)

```

/* Função funcionando */

/*deverá ser aplicado ORCAMENTO o novo valor*/

/*Criar nova coluna na tabela ORCAMENTO */

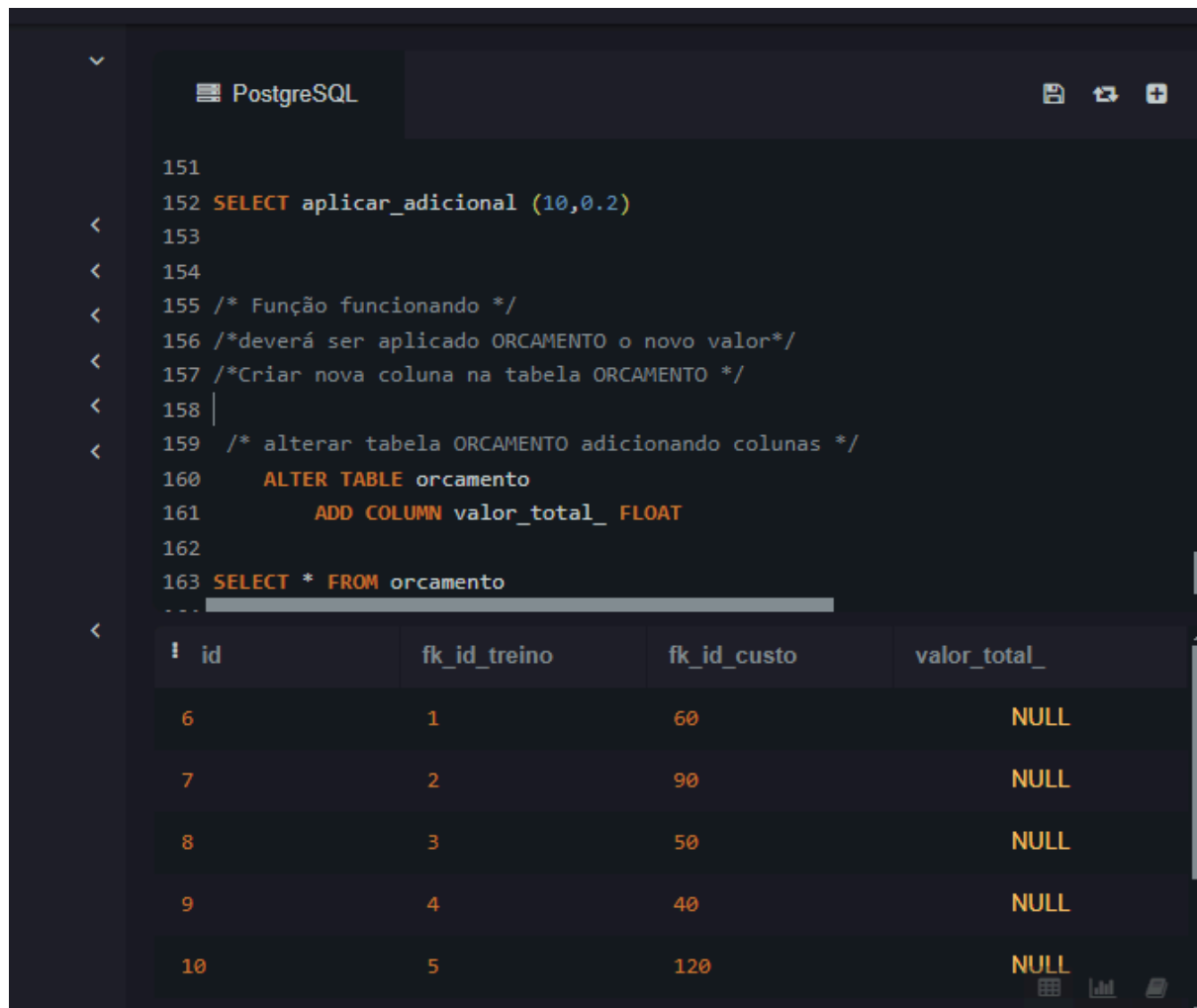
/* alterar tabela ORCAMENTO adicionando coluna_valor TOTAL */

```

ALTER TABLE orcamento
  ADD COLUMN valor_total_ FLOAT

```

SELECT * FROM orcamento



The screenshot shows a PostgreSQL IDE interface. The top panel displays SQL code with line numbers 151 to 163. The code includes a function call, several comments, and an ALTER TABLE statement to add a new column. The bottom panel shows a table view with 5 rows and 4 columns.

```
151
152 SELECT aplicar_adicional (10,0.2)
153
154
155 /* Função funcionando */
156 /*deverá ser aplicado ORCAMENTO o novo valor*/
157 /*Criar nova coluna na tabela ORCAMENTO */
158 |
159 /* alterar tabela ORCAMENTO adicionando colunas */
160     ALTER TABLE orcamento
161         ADD COLUMN valor_total_ FLOAT
162
163 SELECT * FROM orcamento
---
```

id	fk_id_treino	fk_id_custo	valor_total_
6	1	60	NULL
7	2	90	NULL
8	3	50	NULL
9	4	40	NULL
10	5	120	NULL

/* Função funcionando */

/*deverá ser aplicado ORCAMENTO o novo valor*/

/*Criar nova coluna na tabela ORCAMENTO */

/* alterar tabela ORCAMENTO adicionando coluna de Valor Total */

ALTER TABLE orcamento

ADD COLUMN valor_total_ FLOAT

SELECT * FROM orcamento

SELECT * FROM custo

SELECT fk_id_treino, aplicar_adicional(fk_id_custo, 0.20) AS novo_preco FROM orcamento

```
159  /* alterar tabela ORCAMENTO adicionando coluna de Valor Total */
160      ALTER TABLE orcamento
161          ADD COLUMN valor_total_ FLOAT
162
163  SELECT * FROM orcamento
164  SELECT * FROM custo
165
166  SELECT fk_id_treino, aplicar_adicional(fk_id_custo, 0.20) AS novo_preco FROM
167
168
169
170
171 |
```

! fk_id_treino	novo_preco
1	72
2	108
3	60
4	48
5	144

```
< 163 SELECT * FROM orcamento
< 164 SELECT * FROM custo
< 165
< 166 SELECT fk_id_treino, aplicar_adicional(fk_id_custo, 0.20) AS novo_preco
167     FROM orcamento
168     WHERE fk_id_treino=1
169 |
170
171
172
173
```

! fk_id_treino	novo_preco
1	72

/* Agora, só aplicar o Adicional de 20%, nas linhas onde Haverá treino no Fim de Semana = FimSemana = TRUE */

**UPDATE orcamento
SET valor_total_ = aplicar_adicional(100,0.20),
fimsemana =TRUE
WHERE id=6**

**UPDATE orcamento
SET valor_total_ = 100,
fimsemana =FALSE
WHERE id=7**

**UPDATE orcamento
SET valor_total_ = 50,
fimsemana =FALSE
WHERE id=8**

**UPDATE orcamento
SET valor_total_ = aplicar_adicional(150,0.20),
fimsemana =TRUE
WHERE id=9**

**UPDATE orcamento
SET valor_total_ = 200,
fimsemana =FALSE
WHERE id=10**

SELECT * FROM orcamento

```
205     WHERE ID=8
206
207 UPDATE orcamento
208     SET valor_total_ = aplicar_adicional(150,0.20),
209     fimsemana =TRUE
210     WHERE ID=9
211
212 UPDATE orcamento
213     SET valor_total_ = 200,
214     fimsemana =FALSE
215     WHERE ID=10
216
217 SELECT * FROM orcamento
```

id	fk_id_treino	fk_id_custo	valor_total_	fimsemana
6	1	60	120	t
7	2	90	100	f
8	3	50	50	f
9	4	40	180	t
10	5	120	200	f

Finalizando,
reescrevendo os itens das colunas para melhor entendimento */

```
SELECT fk_id_treino AS Ident_Treino, fk_id_custo as Tempo_treino,  
valor_total_ AS Custo_Final FROM orcamento
```

```

216
217 SELECT fk_id_treino AS Ident_Treino, fk_id_custo AS Tempo_treino,
218 valor_total_ AS Custo_Final FROM orcamento
219

```

ident_treino	tempo_treino	custo_final
1	60	120
2	90	100
3	50	50
4	40	180
5	120	200

```

/* =====*/

```

Código total no SQLiteonline

/* criar a tabela CUSTO com os seus atributos */

```

CREATE TABLE custo (
tempo INT NOT NULL PRIMARY KEY,
valor_treino FLOAT NOT NULL,
fimSemana BOOLEAN NOT NULL /* 1=> Fim de semana , 0=> durante a semana */
)
;

```

/* criar a tabela CLIENTE com os seus atributos */

```

CREATE TABLE cliente (
id SERIAL PRIMARY KEY,
nome VARCHAR(48) NOT NULL,
rg VARCHAR(10) NOT NULL,
telefone VARCHAR (12) NOT NULL
);

```

```

CREATE TABLE treino (
/*id SERIAL PRIMARY KEY,*/
treino VARCHAR(48) NOT NULL,

```



```
dia VARCHAR(10) NOT NULL,  
horario VARCHAR (12) NOT NULL,  
professor VARCHAR (12) NOT NULL  
);
```

```
DROP TABLE treino
```

```
CREATE TABLE treino (  
id SERIAL PRIMARY KEY,  
treino VARCHAR(48) NOT NULL,  
dia VARCHAR(10) NOT NULL,  
horario VARCHAR (12) NOT NULL,  
professor VARCHAR (12) NOT NULL  
);
```

```
/*inserir tabela do CLIENTE*/
```

```
INSERT INTO cliente (nome, rg, telefone) VALUES  
('Paulo','353536767', '0645465454'),  
('Tiago','8787653-SP', '0518484949'),  
('Tadeu','535557-RJ', '021567482'),  
('Ronaldo','33646','(021)5657948'),  
('Marcelo','5353535-SP','(021)955555')
```

```
SELECT * from cliente
```

```
/* Inserir dados na tabela TREINO */
```

```
INSERT INTO treino (treino, dia, horario,professor) VALUES  
('cross','01/01/24', '05h00','Alice'),  
('musculacao','05/02/24', '07h00','José'),  
('fitness','11/03/24', '08h00','João'),  
('yoga','12/04/24','10h00','Alex'),  
('Pilatos','04/05/24','11h00','Gabi')
```

```
SELECT * FROM treino
```

```
/* Agora como funcionaria a nossa tabela auxiliar */
```

```
/* Associar o TREINO com CLIENTE */
```

```
/* Criar a tabela EXERCICIO*/
```

```
/* Inserir dados na tabela CUSTO*/
```

```
INSERT INTO custo (tempo,valor_treino,fimSemana) VALUES  
(60,100.00,TRUE),  
(90,100.00,FALSE),  
(50,50.00,False),  
(40,150.00,true),
```

(120,200.00,FALSE)

SELECT * FROM custo

```
/* Agora como funcionaria a nossa tabela auxiliar */
/* Associar o TREINO com CLIENTE */
/* Criar a tabela EXERCICIO*/
CREATE TABLE exercicio (
    id SERIAL PRIMARY KEY,
    fk_id_treino INT NOT NULL,
    fk_id_cliente INT NOT NULL,
    CONSTRAINT fk_treino FOREIGN KEY (fk_id_treino) REFERENCES treino(ID),
    CONSTRAINT fk_cliente FOREIGN KEY (fk_id_cliente) REFERENCES cliente(ID)
);
```

SELECT * FROM exercicio

```
/* Inserindo valores na tabela auxiliar EXERCICIO */
INSERT INTO exercicio (fk_id_treino,fk_id_cliente) VALUES
    (1,1),
    (2,2),
    (3,3),
    (4,4),
    (5,5)
```

SELECT * from exercicio

```
/* Fazendo relacionamentos Tabela CLIENTE e tabela TREINO*/
/* selecionar uma tabela com os registros da tabela TREINO */
/* e com os registros em comum da tabela CLIENTE */
```

```
SELECT * FROM treino /* tabela á esquerda*/
    inner join cliente /* pegar todos os registros em comum entre tabelas */
    on treino.id = cliente.id /* comando on é tipo indicar qual a conexão entre colunas das
tabelas precisam ser iguais */
```

```
/* Agora como funcionaria a nossa tabela auxiliar */
/* Associar o TREINO com CUSTO */
/* Criar a tabela ORCAMENTO*/
CREATE TABLE orcamento (
    id SERIAL PRIMARY KEY,
    fk_id_treino INT NOT NULL,
    fk_id_custo INT NOT NULL,
    CONSTRAINT fk_id_treino FOREIGN KEY (fk_id_treino) REFERENCES treino(ID),
```

```
CONSTRAINT fk_id_custo FOREIGN KEY (fk_id_custo) REFERENCES  
custo(tempo)  
);
```

```
/*DROP TABLE orcamento*/
```

```
/* Inserir valores na tabela ORCAMENTO */
```

```
INSERT INTO orcamento (fk_id_treino,fk_id_custo) VALUES  
    (1,60),  
    (2,90),  
    (3,50),  
    (4,40),  
    (5,120)
```

```
SELECT * FROM orcamento
```

```
/*Fazendo relacionamentos Tabela CUSTO e tabela ORCAMENTO */
```

```
/* selecionar uma tabela com os registros da tabela ORCAMENTO */
```

```
/* e com os registros em comum da tabela CUSTO */
```

```
SELECT * FROM orcamento /* tabela á esquerda*/
```

```
inner join custo /* pegar todos os registros em comum entre tabelas */
```

```
ON orcamento.fk_id_custo = custo.tempo
```

```
/* escrever uma função Aplicar_adicional 20%*/
```

```
CREATE or REPLACE FUNCTION aplicar_adicional (preco FLOAT, adicional FLOAT)  
RETURNS FLOAT as $$
```

```
BEGIN
```

```
RETURN preco * (1 + adicional);
```

```
END
```

```
$$ LANGUAGE PLPGSQL /*PROCEDURAL LANGUAGE postgresql */
```

```
SELECT aplicar_adicional (10,0.2)
```

```
/* Função funcionando */
```

```
/*deverá ser aplicado ORCAMENTO o novo valor*/
```

```
/*Criar nova coluna na tabela ORCAMENTO */
```

```
/* alterar tabela ORCAMENTO adicionando coluna de Valor Total */
```

```
ALTER TABLE orcamento
```

```
ADD COLUMN valor_total_ FLOAT
```

```
SELECT * FROM orcamento
```

```
SELECT * FROM custo
```

```
SELECT fk_id_treino, aplicar_adicional(fk_id_custo, 0.20) AS novo_preco
FROM orcamento
WHERE fk_id_treino=1
```

```
/*-----*/
```

```
SELECT * from custo
```

```
/* Aplicar Adicional somente onde fimSemana=TRUE */
```

```
ALTER TABLE orcamento
add COLUMN fimsemana BOOLEAN;
```

```
SELECT * FROM orcamento
```

```
UPDATE orcamento
SET fimsemana = TRUE
```

```
UPDATE orcamento
set fimsemana = FALSE
```

```
UPDATE orcamento
SET valor_total_ = 100,
fimsemana =TRUE
WHERE id=6
```

```
UPDATE orcamento
SET valor_total_ = aplicar_adicional(100,0.20),
fimsemana =TRUE
WHERE id=6
```

```
UPDATE orcamento
SET valor_total_ = 100,
fimsemana =FALSE
WHERE id=7
```

```
UPDATE orcamento
SET valor_total_ = 50,
fimsemana =FALSE
WHERE id=8
```

```
UPDATE orcamento
SET valor_total_ = aplicar_adicional(150,0.20),
fimsemana =TRUE
WHERE id=9
```

```
UPDATE orcamento
SET valor_total_ = 200,
```

```
fimsemana =FALSE  
WHERE id=10
```

```
SELECT fk_id_treino AS Ident_Treino, fk_id_custo as Tempo_treino,  
valor_total_ AS Custo_Final FROM orcamento
```

```
=====
```