

Fit CIn

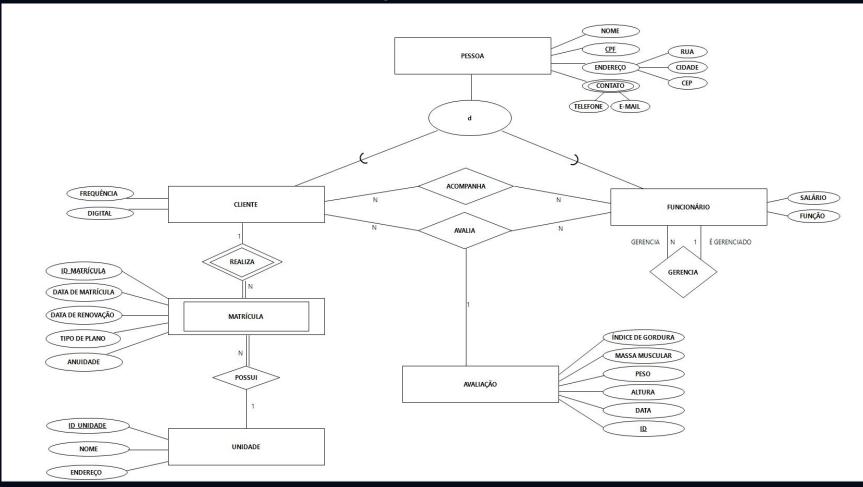
Dara Caroline de Souza Vasconcelos - dcsv Edinaldo Barbosa da Costa Filho - ebcf2 João Pedro Souza Pereira de Moura - jpspm Maria Eduarda Souza Oliveira de Senna mesos

Rafael Jesus de Araujo Vasconcelos - rjav

Mini Mundo

Um sistema de uma rede de academias, "Fit Cln", há pessoas que são registradas contendo: CPF (Identificador), Nome, Endereço (Rua, Cidade, CEP) e podem ter contato (multivalorado) contendo email e/ou telefone; elas se dividem em 2 categorias : clientes e funcionários. Os atributos dos clientes são: Frequência, Digital. Os atributos dos funcionários são: Salário e Função na academia. Para cada cliente tem unicamente uma matrícula relacionada, constando: Data de Renovação, Tipo de Plano, Anuidade, Data de Matrícula. Funcionários acompanham clientes durante o treino, mas podem gerenciar outros funcionários. Um único funcionário pode acompanhar diversos clientes e um cliente pode ser acompanhado por diversos funcionários. Um cliente pode ser avaliado por um ou mais funcionários, e um funcionário pode avaliar um ou mais clientes, gerando avaliações contendo índice de gordura, massa muscular, peso, altura e data como atributos.

Projeto Conceitual



Projeto Lógico

```
Pessoa (CPF, nome, end rua, end cidade, end CEP);
Contato(CPF, telefone, email)
     CPF \rightarrow Pessoa(CPF)
Avaliacao(ID, data, altura, peso, massa muscular, indice de gordura)
Unidade(ID unidade, nome, endereco)
Funcionario(CPF, funcao, salario)
     CPF \rightarrow Pessoa(CPF)
Gerencia(gerente!, gerenciado)
      gerente → Funcionario(CPF)
      gerenciado → Funcionario(CPF)
Cliente(CPF, frequencia, digital)
     CPF \rightarrow Pessoa(CPF)
Matricula (CPF,ID matricula, data de matricula, tipo de plano, anuidade, data de renovação,
ID unidade!)
     \overline{\mathsf{CPF}} \to \mathsf{Cliente}(\overline{\mathsf{CPF}})
     ID unidade→ Unidade(ID unidade)
Avalia (CPF funcionario, CPF aluno, ID!)
     CPF Funcionario → Funcionario (CPF)
     CPF Aluno → Cliente(CPF)
Acompanha(acompanhante, acompanhado)
     acompanhante → Funcionario(CPF)
     acompanhado → Cliente(CPF)
```

--GROUP BY / HAVING

SELECT ACAD.end_cidade, ACAD.nome,

COUNT(*) AS Quantidade

FROM matricula M, unidade ACAD

WHERE M.id_unidade = ACAD.id_unidade

GROUP BY ACAD.end_cidade, ACAD.nome

HAVING COUNT(*) > 1;

--INNER JOIN

SELECT nome, funcao
FROM pessoa P INNER JOIN
funcionario F ON P.cpf = F.CPF;

--OUTER JOIN

SELECT P.nome, email

FROM pessoa P

LEFT OUTER JOIN contato C on P.cpf = C.cpf;

--Semi junção

```
SELECT M.cpf, P.nome
FROM matricula M, pessoa P
WHERE EXISTS (SELECT * FROM cliente C WHERE M.cpf = C.cpf AND P.cpf =
M.cpf AND C.frequencia > 5);
```

--Anti-junção

SELECT P.nome, P.cpf FROM pessoa P
WHERE P.cpf NOT IN (SELECT C.cpf FROM contato C

WHERE C.telefone IS NULL);

--subconsulta escalar

```
SELECT UNI.NOME
FROM unidade UNI
WHERE (SELECT AVG(peso)
     FROM matricula M, avaliacao AV, avalia A
     WHERE M.cpf = A.cpf_aluno AND AV.id_avaliacao = A.id_avaliacao
     AND M.id unidade = UNI.id unidade) > 75;
```

--subconsulta do tipo linha

SELECT M.cpf FROM matricula M
WHERE (M.id_unidade, M.tipo_plano) = (SELECT MATRI.id_unidade,
MATRI.tipo plano

FROM matricula MATRI
WHERE MATRI.cpf = '76454768007');

--subconsulta do tipo tabela

```
SELECT M.cpf
FROM matricula M
WHERE M.CPF IN (SELECT C.cpf
               FROM cliente C
               WHERE C.frequencia > (SELECT
AVG (frequencia)
                                         FROM
cliente));
```

--operação de conjunto

FROM unidade A;

SELECT P.end_cidade AS CIDADE, 'Pessoa' AS TIPO
FROM pessoa P
UNION
SELECT A.end cidade AS CIDADE, 'Unidade' AS TIPO

Procedimentos

END;

```
CREATE OR REPLACE PROCEDURE print nome (auxCPF
                                                   BEGIN
                                                   print nome('74055189079');
VARCHAR) IS auxNome VARCHAR(50);
                                                   END;
BEGIN
   SELECT P.nome INTO auxNome
   FROM pessoa P
   WHERE p.cpf = auxCPF;
   dbms output.put line(auxNome);
   EXCEPTION
       WHEN NO DATA FOUND THEN
       dbms output.put line('CPF NOT FOUND');
```

Procedimentos

```
CREATE OR REPLACE PROCEDURE city(cidade VARCHAR) IS
CURSOR nomes IS
   SELECT nome
   FROM PESSOA
   WHERE end cidade = cidade;
aux VARCHAR (50);
BEGIN
   OPEN nomes;
       LOOP
           FETCH nomes INTO aux;
           EXIT WHEN nomes % NOTFOUND;
           dbms output.put line(aux);
       END LOOP;
   CLOSE nomes;
END;
```

```
BEGIN
city('Recife');
END;
```

Função

RETURN 'Funcionario não cadastrado';

```
BEGIN
CREATE OR REPLACE FUNCTION get funcao(auxCPF
                                               dbms output.put line(get funcao('74055189079'))
VARCHAR) RETURN VARCHAR IS auxFuncao
VARCHAR(20);
                                               END;
BEGIN
   SELECT F.funcao INTO auxFuncao
                                               BEGIN
   FROM funcionario F
                                               dbms output.put line(get funcao('74055189079'))
   WHERE F.cpf = auxCPF;
                                               END;
   RETURN auxFuncao;
   EXCEPTION
      WHEN NO DATA FOUND THEN
```

Gatilho

CREATE OR REPLACE TRIGGER indicador membro

```
AFTER INSERT ON pessoa
REFERENCING NEW AS P
FOR EACH ROW
BEGIN
    dbms output.put line('O membro ' || :P.nome || ' FOI CADASTRADA
COM SUCESSO');
END;
INSERT INTO pessoa
VALUES (74055184579, 'Aroldo', 'Avenida Chaves', 'Teresina', 64071250);
```

Gatilho

FOR EACH ROW

CREATE OR REPLACE TRIGGER indicador_academia



```
AFTER INSERT ON unidade REFERENCING NEW AS ACAD
```

```
BEGIN
    dbms_output.put_line('A ACADEMIA ' || :ACAD.nome || ' FOI
REGISTRADA COM SUCESSO');
END;
INSERT INTO unidade
VALUES(00100, 'DO FINO PRO GRANDE', 'Rua das Flores', 'Recife',
52071250);
```