

Projeto - Gestão de agend. de Lab. CIn

Fernanda de Araújo Lima Pascoal,
Gabriel Laroche Borba,
Maria Geyzianny de Sousa Silva,
Valter José da Silva Junior;



Centro de
Informática
UFPE



UNIVERSIDADE
FEDERAL
DE PERNAMBUCO

Projeto Conceitual



Projeto Lógico

cracha(num_cracha, data_emissao)

usuario(matricula, nome!, email!, senha!, [num_cracha]!)

telefone(matricula, telefone)

matricula -> usuario(matricula)

num_cracha -> cracha(num_cracha)

agendamento (cod_agend, data_inicio, data_fim)

laboratorio (cod_lab, endereco_num_sala, endereco_predio)

equipamento(id_equipamento, cod_lab)

cod_lab -> laboratorio(cod_lab)

peca(cod_peca, qtd)

visitante (matricula, id_visitante, motivo_visita)

matricula -> usuário(matricula)

professor(matricula, cadeiras, prof_coordenador!)

matricula -> usuario(matricula)

prof_coordenador -> professor(matricula)

aluno(matricula, periodo, cadeiras)

matricula -> usuario(matricula)

funcionario_manutencao(matricula, salário,

num_carteira)

matricula -> usuario(matricula)

manutencao(matricula, id_equipamento,

data_inicio_manutencao, data_fim_manutencao,

descricao)

matricula -> funcionario_manutencao(matricula)

id_equipamento -> Equipamento(id_equipamento)

troca(matricula, id_equipamento, cod_peca)

matricula, id_equipamento -> manutencao(matricula,

id_equipamento)

cod_peca -> Peca(cod_peca)

acessa(matricula!, cod_agend, cod_lab)

matricula -> usuario(Matricula)

cod_agend -> agendamento(cod_agend)

cod_lab -> laboratório(cod_lab)

Projeto Físico

```
CREATE TABLE CRACHA (  
  num_cracha NUMBER(4),  
  data_emissao DATE,  
  CONSTRAINT PK_CRACHA PRIMARY KEY(num_cracha)  
);
```

```
CREATE TABLE USUARIO (  
  matricula NUMBER(10),  
  nome VARCHAR2(80) NOT NULL,  
  email VARCHAR2(100) NOT NULL,  
  senha VARCHAR2(100) NOT NULL,  
  num_cracha NUMBER(4) NOT NULL UNIQUE,  
  CONSTRAINT PK_USUARIO PRIMARY KEY (matricula),  
  CONSTRAINT FK_USU_CRA FOREIGN KEY (num_cracha)  
  REFERENCES CRACHA (num_cracha)  
);
```

```
CREATE TABLE TELEFONES (  
  matricula NUMBER(10),  
  telefone VARCHAR2(20),  
  CONSTRAINT PK_TELEFONES PRIMARY KEY (matricula,  
  telefone),  
  CONSTRAINT FK_TEL_USU FOREIGN KEY (matricula)  
  REFERENCES USUARIO (matricula)  
);
```

```
CREATE TABLE AGENDAMENTO (  
  cod_agend NUMBER(4),  
  data_inicio DATE NOT NULL,  
  data_fim DATE NOT NULL,  
  CONSTRAINT PK_AGENDAMENTO PRIMARY KEY (cod_agend)  
);
```

```
CREATE TABLE LABORATORIO (  
  cod_lab NUMBER(4),  
  endereco_num_sala VARCHAR2(50),  
  endereco_predio VARCHAR2(50),  
  CONSTRAINT PK_LABORATORIO PRIMARY KEY (cod_lab)  
);  
CREATE TABLE EQUIPAMENTO (  
  id_equipamento NUMBER(4),  
  cod_lab NUMBER(4),  
  CONSTRAINT PK_EQUIPAMENTO PRIMARY KEY  
  (id_equipamento),  
  CONSTRAINT FK_COD_LAB FOREIGN KEY (cod_lab)  
  REFERENCES LABORATORIO(cod_lab)  
);
```

```
CREATE TABLE PECA (  
  cod_pecas NUMBER(4),  
  qtd NUMBER(4) NOT NULL,  
  CONSTRAINT PK_PECA PRIMARY KEY (cod_pecas)  
);
```

```
CREATE TABLE VISITANTE (  
  matricula NUMBER(10),  
  id_visitante VARCHAR2(10),  
  motivo_visita VARCHAR2(100),  
  CONSTRAINT PK_VISITANTE PRIMARY KEY(matricula, id_visitante),  
  CONSTRAINT FK_VIS_USU FOREIGN KEY (matricula) REFERENCES  
  USUARIO(matricula)  
);
```

Projeto Físico

```
CREATE TABLE PROFESSOR (  
  matricula NUMBER(10),  
  cadeiras VARCHAR2(100),  
  prof_coordenador NUMBER(10) NOT NULL,  
  CONSTRAINT PK_PROFESSOR PRIMARY KEY  
  (matricula),  
  CONSTRAINT FK_PRO_USU FOREIGN KEY (matricula)  
  REFERENCES USUARIO (matricula),  
  CONSTRAINT FK_PRO_PROF FOREIGN KEY  
  (prof_coordenador) REFERENCES PROFESSOR  
  (matricula)  
);
```

```
CREATE TABLE ALUNO (  
  matricula NUMBER(10),  
  periodo VARCHAR2(50),  
  cadeira VARCHAR2(100),  
  CONSTRAINT PK_ALUNO PRIMARY KEY (matricula),  
  CONSTRAINT FK_ALU_USU FOREIGN KEY (matricula)  
  REFERENCES USUARIO (matricula)  
);
```

```
CREATE TABLE FUNCIONARIO_MANUTENCAO (  
  matricula NUMBER(10),  
  salario NUMBER(4),  
  num_carteira VARCHAR2(50),  
  CONSTRAINT PK_FUNC_MANUT PRIMARY KEY (matricula),  
  CONSTRAINT FK_FUNC_MANUT_USU FOREIGN KEY (matricula)  
  REFERENCES USUARIO (matricula)  
);
```

```
CREATE TABLE MANUTENCAO (  
  matricula NUMBER(10),  
  id_equipamento NUMBER(4),  
  data_inicio_manutencao DATE,  
  data_fim_manutencao DATE,  
  descricao VARCHAR2(200),  
  CONSTRAINT PK_MANUTENCAO PRIMARY KEY (matricula,  
  id_equipamento),  
  CONSTRAINT FK_MANU_EQUIP FOREIGN KEY (id_equipamento)  
  REFERENCES EQUIPAMENTO (id_equipamento),  
  CONSTRAINT FK_MANU_FUNC_MANUT FOREIGN KEY (matricula)  
  REFERENCES FUNCIONARIO_MANUTENCAO (matricula)  
);
```

```
CREATE TABLE TROCA (  
  matricula NUMBER(10),  
  id_equipamento NUMBER(4),  
  cod_pecas NUMBER(4),  
  CONSTRAINT PK_TROCA PRIMARY KEY (matricula, id_equipamento,  
  cod_pecas),  
  CONSTRAINT FK_TROCA_MANU_MAT FOREIGN KEY (matricula,  
  id_equipamento) REFERENCES MANUTENCAO (matricula, id_equipamento),  
  CONSTRAINT FK_TROCA_PECAS FOREIGN KEY (cod_pecas) REFERENCES  
  PECAS (cod_pecas)  
);
```

```
CREATE TABLE TROCA (  
  matricula NUMBER(10),  
  id_equipamento NUMBER(4),  
  cod_pecas NUMBER(4),  
  CONSTRAINT PK_TROCA PRIMARY KEY (matricula, id_equipamento,  
  cod_pecas),  
  CONSTRAINT FK_TROCA_MANU_MAT FOREIGN KEY (matricula,  
  id_equipamento) REFERENCES MANUTENCAO (matricula, id_equipamento),  
  CONSTRAINT FK_TROCA_PECAS FOREIGN KEY (cod_pecas) REFERENCES  
  PECAS (cod_pecas)  
);
```

Consulta - Junção Interna

- Mostre o nome e data emissão do cracha de todos os usuarios que são alunos e estão no segundo período

```
1  -- CONSULTAS SELECT
2
3  -- Mostre o nome e data emissão do cracha de todos os usuarios que são alunos e estão no segundo período
4  SELECT U.NOME, C.DATA_EMISSAO
5  FROM USUARIO U INNER JOIN CRACHA C ON U.NUM_CRACHA = C.NUM_CRACHA INNER JOIN ALUNO A ON U.MATRICULA = A.MATRICULA
6  WHERE PERIODO = '2';
7
8
9
10
11
12
13
14
15
```

NOME	DATA_EMISSAO
Valter Junior	04-APR-22
Vinicius Monitor	05-MAY-22

Consulta - Junção Interna com Subconsulta Tabela

- Deve-se mostrar o nome de todos os alunos que agendaram laboratórios na data 02-JAN-22

```
8  -- Deve-se mostrar o nome de todos os alunos que agendaram laboratórios na data 02-JAN-22
9  SELECT U.NOME
10 FROM USUARIO U INNER JOIN ACESSA A
11      ON U.MATRICULA = A.MATRICULA INNER JOIN AGENDAMENTO AG
12      ON A.COD_AGEND = AG.COD_AGEND
13 WHERE AG.DATA_INICIO = '02-JAN-22' AND U.MATRICULA IN (
14     SELECT A.MATRICULA
15     FROM ALUNO A);
16
17
18
19
20
21
22
```

NOME
Vinicius Monitor

Consulta - Group By/ Having

- Mostre os usuários que levaram mais de um 1 visitante

```
21
22
23 -- Mostre os usuarios que levaram mais de um 1 visitante
24 v SELECT U.NOME, COUNT(*)
25 FROM USUARIO U INNER JOIN VISITANTE V ON U.MATRICULA = V.MATRICULA
26 GROUP BY U.NOME
27 HAVING COUNT(*) > 1;
28
29
30
31
32
33
34
35
36
```

NOME	COUNT(*)
Olivia Taylor	3
Sophie Brown	2

Consulta - Junção Externa

- Mostre o Num_Sala dos laboratórios que nunca tiveram um agendamento

```
35
36
37 -- Mostre o Num_Sala dos laboratorios que nunca tiveram um agendamento
38 v SELECT L.ENDereco_NUM_SALA
39 FROM LABORATORIO L LEFT JOIN ACESSA A ON L.COD_LAB = A.COD_LAB
40 WHERE A.COD_LAB IS NULL;
41
42
43
44
45
46
47
48
49
```

ENDereco_NUM_SALA

Lab PET

Consulta - Semi Join

- Mostre o nome e email dos usuários que já levaram algum visitante, não pode fazer a consulta com JOIN

```
49
50 -- Mostre o nome e email dos usuários que já levaram algum visitante, não pode fazer a consulta com JOIN
51 ✓ SELECT U.NOME, U.EMAIL
52 FROM USUARIO U
53 WHERE EXISTS (
54     SELECT *
55     FROM VISITANTE V
56     WHERE V.MATRICULA = U.MATRICULA);
57
58
59
60
61
62
63 |
```

NOME	EMAIL
Olivia Parker	oliviaparker@outlook.com
Sophie Brown	sophiebrown@gmail.com
Adam Scott	adamscott@yahoo.com
Olivia Taylor	oliviataaylor@hotmail.com

Consultas - Anti Join

- Mostre o nome e o número do crachá dos usuários que nunca levaram algum visitante, não pode fazer a consulta com JOIN, e faça ordenado pelo número do crachá

```
62
63
64 -- Mostre o nome e o número do crachá dos usuários que nunca levaram algum visitante, não pode fazer a consulta com JOIN, e faça ordenado pelo número do cracha
65 ✓ SELECT U.NOME, U.NUM_CRACHA
66 FROM USUARIO U
67 WHERE NOT EXISTS (
68     SELECT *
69     FROM VISITANTE V
70     WHERE V.MATRICULA = U.MATRICULA)
71 ORDER BY U.NUM_CRACHA;
72
73
74
75
76
```

NOME	NUM_CRACHA
Maria Geyzianny	1
Fernanda Pascoal	2
Gabriel Laroche	3

Consultas - Subconsulta Escalar

- Mostre o código das peças que tem mais ou um número igual de peças que a média total de todos os tipos de peças

```
76  
77  
78 -- Mostre o código das peças que tem mais ou um número igual de peças que a média total de todos os tipos de peças  
79 v SELECT P.COD_Peca  
80 FROM Peca P  
81 WHERE P.QTD >= (SELECT AVG(QTD)  
82 FROM Peca)  
83  
84  
85  
86  
87  
88  
89  
90
```

COD_Peca
2
3

Consultas - Subconsulta Linha

- Mostre a Nome dos alunos que cursam a mesma cadeira e estão no mesmo período do aluno com matrícula 12350

```
90
91 -- Mostre a Nome dos alunos que cursam a mesma cadeira e estão no mesmo período do aluno com matrícula
92 SELECT U.NOME
93 FROM ALUNO A, USUARIO U
94 WHERE A.MATRICULA = U.MATRICULA AND
95       (A.CADEIRA, A.PERIODO) = (
96       SELECT A1.CADEIRA, A1.PERIODO
97       FROM ALUNO A1
98       WHERE A1.MATRICULA = 12350
99       AND A.MATRICULA <> 12350
100 );
101
102
103
104
```

NOME
Fernanda Pascoal
Miguel Diaz

Download CSV

Consultas - Subconsulta Tabela

- Mostre a matrícula dos funcionários que começaram e terminaram uma manutenção no mesmo dia do funcionário com matrícula 12356

```
103
104
105 -- Mostre a matrícula dos funcionários que começaram e terminaram uma manutenção no mesmo dia do funcionário com matrícula 12356
106 v SELECT M1.MATRICULA
107 FROM MANUTENCAO M1
108 WHERE (M1.DATA_INICIO_MANUTENCAO, M1.DATA_FIM_MANUTENCAO) IN (
109     SELECT M2.DATA_INICIO_MANUTENCAO, M2.DATA_FIM_MANUTENCAO
110     FROM MANUTENCAO M2
111     WHERE M2.MATRICULA = 12356
112     AND M1.MATRICULA <> 12356);
113
114
115
116
117
118
```

MATRICULA
12358
12357

Consultas - Operação de conjunto - INTERSECT

- Mostre os usuários que já levaram algum visitante, usando intersect

```
1  -- Mostre os usuários que já levaram algum visitante, usando intersect
2  v SELECT MATRICULA
3      FROM USUARIO
4      INTERSECT
5      (SELECT MATRICULA
6       FROM VISITANTE)
7
```

MATRICULA
12351
12357
12358
12359

Procedimentos

- Retorna a quantidade total de Laboratórios

```
1  -- Retorna a qauntidade total de Laboratórios
2  v CREATE OR REPLACE
3  PROCEDURE
4      exhibir_qtd_lab
5  IS
6      qtdLab number;
7  v BEGIN
8      SELECT COUNT(COD_LAB) INTO qtdLab
9      FROM LABORATORIO;
10     DBMS_OUTPUT.PUT_LINE('Quantidade : '||qtdLab);
11 END;
12
13 v BEGIN
14     exhibir_qtd_lab;
15 END;
```

Statement processed.
Quantidade : 5

Funções

- Retorna qual o período em que um aluno está

```
2 v CREATE OR REPLACE FUNCTION
3   mostra_perodo_aluno(codU IN NUMBER)
4   RETURN VARCHAR2
5 IS
6   v_perodo VARCHAR2(40);
7 v BEGIN
8   SELECT PERIODO INTO v_perodo
9   FROM ALUNO
10  WHERE MATRICULA = codU;
11
12   RETURN v_perodo;
13 v EXCEPTION
14   WHEN NO_DATA_FOUND THEN
15     RETURN 'Aluno não foi encontrado';
16 END;
```

Statement processed.
Período : 4

```
18
19
20
21
22
23
24
25
26
27 v DECLARE
28   periodo VARCHAR2(40);
29 v BEGIN
30   periodo := mostra_perodo_aluno(12360);
31   DBMS_OUTPUT.PUT_LINE('Período : '||periodo);
32 END;
```

Statement processed.
Período : 4

Procedimentos

- Altera o email cadastrado de um usuário

```
1  -- Altera o email cadastrado de um usuário
2  v CREATE OR REPLACE
3  PROCEDURE
4      altera_email(
5      codU NUMBER,
6      novo_email USUARIO.EMAIL%TYPE
7      )
8  IS
9  BEGIN
10     UPDATE USUARIO
11     SET EMAIL = novo_email
12     WHERE MATRICULA = codU;
13     COMMIT;
14 END;
15
```

```
15
16 v BEGIN
17     altera_email(1234, 'glarocheborba@gmail.com');
18 END;
19
20 SELECT * FROM USUARIO;
```

MATRICULA	NOME	EMAIL	SENHA	NUM_CRACHA
12345	Paulo Vitor	paulo@gmail.com	pass444	6
12346	Fernanda Pascoal	glarocheborba@gmail.com	pass456	2
12347	Gabriel Laroche	glarocheborba@gmail.com	secure789	3

Funções

- Deve - se mostrar os laboratórios disponíveis.

```
46
47 SELECT * FROM AGENDAMENTO A;
```

COD_AGEND	DATA_INICIO	DATA_FIM
1	02-JAN-22	03-JAN-22
2	03-FEB-22	04-FEB-22
3	04-MAR-22	05-MAR-22
4	05-APR-22	06-APR-22

```
46
47 SELECT * FROM LABORATORIO L;
```

COD_LAB	ENDERECO_NUM_SALA	ENDERECO_PREDIO	ID_EQUIPA
1	GRAD4	Laboratório 4	—
2	GRAD1	Laboratório 1	9
3	GRAD2	Laboratório 2	7
4	GRAD3	Laboratório 3	3
5	Lab PET	Laboratório do PET	4

Funções

- Deve - se mostrar os laboratórios disponíveis.

```
CREATE OR REPLACE FUNCTION verificar_laboratorios_disponiveis (  
    p_data IN DATE  
)  
RETURN SYS_REFCURSOR  
IS  
    c_resultados SYS_REFCURSOR;
```

```
BEGIN  
    OPEN c_resultados FOR  
        SELECT L.cod_lab, L.endereco_num_sala, L.endereco_predio,  
            CASE  
                WHEN EXISTS (  
                    SELECT *  
                    FROM ACESSA A  
                    INNER JOIN AGENDAMENTO AG ON A.cod_agend = AG.cod_agend  
                    WHERE A.cod_lab = L.cod_lab  
                    AND (  
                        AG.data_inicio BETWEEN p_data AND p_data+1  
                        OR AG.data_fim BETWEEN p_data AND p_data+1  
                        OR p_data BETWEEN AG.data_inicio AND AG.data_fim  
                        OR p_data+1 BETWEEN AG.data_inicio AND AG.data_fim  
                    )  
                )  
            THEN 'Laboratório indisponível para o período especificado.'
```

Funções

- Deve - se mostrar os laboratórios disponíveis.

```
        )  
        ) THEN 'Laboratório indisponível para o período especificado.'  
        ELSE 'Laboratório disponível para o período especificado.'  
    END AS disponibilidade  
FROM LABORATORIO L;  
  
RETURN c_resultados;  
END;
```

Funções

- Deve - se mostrar os laboratórios disponíveis.

DECLARE

```
c SYS_REFCURSOR;  
v_cod_lab LABORATORIO.cod_lab%TYPE;  
v_endereco_num_sala LABORATORIO.endereco_num_sala%TYPE;  
v_endereco_predio LABORATORIO.endereco_predio%TYPE;  
v_disponibilidade VARCHAR2(100);
```

BEGIN

```
c := verificar_laboratorios_disponiveis(TO_DATE('2022-01-02', 'YYYY-MM-DD'));
```

LOOP

```
    FETCH c INTO v_cod_lab, v_endereco_num_sala, v_endereco_predio, v_disponibilidade;
```

```
    EXIT WHEN c%NOTFOUND;
```

```
    DBMS_OUTPUT.PUT_LINE
```

```
        (v_cod_lab || ' - ' || v_endereco_num_sala || ', ' || v_endereco_predio || ' - ' || v_disponibilidade);
```

```
END LOOP;
```

```
CLOSE c;
```

```
END.
```

Funções

- Deve - se mostrar os laboratórios disponíveis.

```
35     v_disponibilidade VARCHAR2(100);
36 BEGIN
37     c := verificar_laboratorios_disponiveis(TO_DATE('2022-01-02', 'YYYY-MM-DD'));
38     LOOP
39         FETCH c INTO v_cod_lab, v_endereco_num_sala, v_endereco_predio, v_disponibilidade;
40         EXIT WHEN c%NOTFOUND;
41         DBMS_OUTPUT.PUT_LINE
42             (v_cod_lab || ' - ' || v_endereco_num_sala || ', ' || v_endereco_predio || ' - ' || v_disponibilidade);
43     END LOOP;
44     CLOSE c;
45 END;
```

Statement processed.

- 1 - GRAD4, Laboratório 4 - Laboratório indisponível para o período especificado.
- 2 - GRAD1, Laboratório 1 - Laboratório disponível para o período especificado.
- 3 - GRAD2, Laboratório 2 - Laboratório disponível para o período especificado.
- 4 - GRAD3, Laboratório 3 - Laboratório disponível para o período especificado.
- 5 - Lab PET, Laboratório do PET - Laboratório disponível para o período especificado.

Obrigada(o).