[](http://portal.anhembi.br/publique/cgi/cgilua.exe/sys/start.htm?tpl=home)

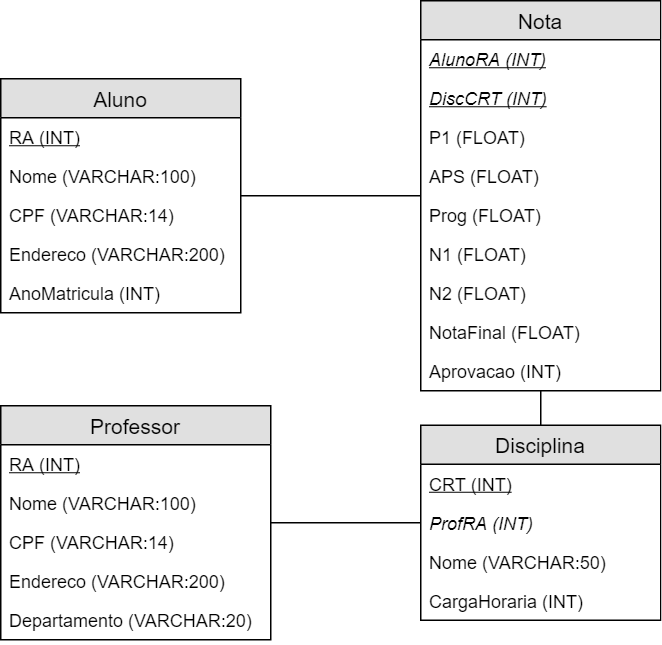
Disciplina: Banco de Dados II

Prof. Bruno Matheus

**Atividade 2: Procedure & Function**

**Introdução:**

O banco de dados deste exercício segue o seguinte Modelo Relacional:



Comandos de criação:

CREATE TABLE Aluno (

RA INT PRIMARY KEY IDENTITY(100,1),

Nome VARCHAR(100),

CPF VARCHAR(14),

Endereco VARCHAR(200),

AnoMatricula INT

)

CREATE TABLE Professor (

RA INT PRIMARY KEY IDENTITY(1000,1),

Nome VARCHAR(100),

CPF VARCHAR(14),

Endereco VARCHAR(200),

Departamento VARCHAR(20)

)

CREATE TABLE Disciplina(

CRT INT PRIMARY KEY IDENTITY(10,1),

ProfRA INT REFERENCES Professor,

Nome VARCHAR(50),

CargaHoraria INT

)

CREATE TABLE Nota(

AlunoRA INT FOREIGN KEY REFERENCES Aluno,

DiscCRT INT FOREIGN KEY REFERENCES Disciplina,

P1 DECIMAL(3,1),

APS DECIMAL(3,1),

Prog DECIMAL(3,1),

N1 DECIMAL(3,1),

N2 DECIMAL(3,1),

NotaFinal DECIMAL(3,1),

Aprovacao INT DEFAULT 0

)

Em seguida, execute os seguintes comandos de inserção:

INSERT INTO Aluno (Nome, CPF, Endereco, AnoMatricula)

VALUES ('Andre', '111.111.111-11', 'Rua A, n 1', 2015),

('Bandre', '222.111.111-11', 'Rua B, n 2', 2016),

('Candre', '333.111.111-11', 'Rua C, n 3', 2017),

('Dandre', '444.111.111-11', 'Rua D, n 4', 2018),

('Eandre', '555.111.111-11', 'Rua E, n 5', 2019)

INSERT INTO Professor (Nome, CPF, Endereco, Departamento)

VALUES ('Zandre', '26.111.111-11', 'Rua Z, n 26', 'Engenharia'),

('Yandre', '25.111.111-11', 'Rua Y, n 25', 'Farmacia'),

('Xandre', '24.111.111-11', 'Rua X, n 24', 'Direito')

INSERT INTO Disciplina (ProfRA, Nome, CargaHoraria)

VALUES (1000, 'Banco de Fichas', 60),

(1000, 'Programação Orientada a Comidas', 50),

(1001, 'Estrutura de Linguiças', 90),

(1001, 'Colchões', 60)

INSERT INTO Nota (DiscCRT, AlunoRA, P1, APS, Prog, N2)

VALUES (10, 100, 5, 8, 6, 5),

(10, 101, 2, Null, 7, 3),

(11, 100, Null, 8, 6, 8),

(11, 101, 6, 4, 7, 5),

(12, 103, 4, 8, Null, 8),

(13, 101, 5, 5, 7, 2),

(13, 104, 8, 9, 10, Null),

(13, 103, 8, 5, 10, 4)

**Atividade:**

Crie os procedures necessários para realizar as operações abaixo:

1. A tabela nota tem uma coluna N1 (FLOAT) que deve ser calculada como N1 = (P1+APS+Prog)/3. Crie um procedure chamado **CalcularNota** que insere os valores corretos de N1 em todas as linhas da tabela Nota. Lembre-se que o comando UPDATE pode alterar várias linhas simultaneamente. Execute o procedure.
2. Modifique o procedure **CalcularNota** acima, para que ele calcule a NotaFinal também, lembrando que NF = (N1\*0,4+N2\*0,6). Execute o procedure.
3. Se você visualizar a sua tabela Nota após executar o procedure **CalcularNota** acima, vai notar que algumas notas N1 e NotaFinal ainda estão com valor Null.
   1. Explique por que isso aconteceu?
   2. Altere o procedure **CalcularNota** acima de modo que ele resolva o problema e todas as notas N1 e NotaFinal recebam valores corretos. Execute o procedure.

**DICA:** O que acontece com a nota de um aluno que não vem fazer uma prova ou não entrega a APS?

1. A tabela nota tem uma coluna chamada Aprovacao (INT), que informa se o aluno está aprovado ou não na disciplina. Crie um novo procedure chamado **CalculaAprovacao** que avalia se o aluno tirou mais do que 5 ou não na sua nota final. Se a nota final for maior ou igual a 5 o aluno deve ter o campo Aprovacao mudado para 1, se não o valor de Aprovacao deve continuar 0. Execute o procedure.
2. Altere o procedure **CalculaAprovacao**, para que ele calcule as notas N1 e NotaFinal antes de verificar aprovação. Lembre-se que um procedure pode chamar outro. Execute o procedure.
3. E se você quiser mudar a nota mínima para aprovação? Altere o procedure **CalculaAprovacao** de modo que ele receba como entrada um valor numérico FLOAT para a nota de aprovação. Execute com o valor 4,5 e veja se algo foi alterado.
4. Crie um procedure chamado ChecaAprovacao, que deve ter como entrada o RA de um aluno e o CRT de uma Disciplina. O Procedure deve retornar em uma variável (*VARCHAR*) o valor ‘Aprovado’ se o aluno foi aprovado e ‘Não aprovado’ se o aluno foi reprovado. Execute o procedure.
5. Refaça o procedure acima (item 7) como uma função (FUNCTION), que deve receber RA e CRT como entrada e imprimir ‘Aprovado’ ou ‘Não aprovado’. Teste a função.
6. O formato da tabela Nota acima consome muito mais memória do que deveria, pois ela guarda valores derivados, ou seja, valores que podem ser calculados usando outras colunas da mesma tabela. Isso é verdade para N1, NotaFinal e Aprovacao. Delete a tabela Nota e reconstrua usando os seguintes comandos:

CREATE TABLE Nota(

AlunoRA INT FOREIGN KEY REFERENCES Aluno,

DiscCRT INT FOREIGN KEY REFERENCES Disciplina,

P1 DECIMAL(3,1),

APS DECIMAL(3,1),

Prog DECIMAL(3,1),

N2 DECIMAL(3,1)

)

INSERT INTO Nota (DiscCRT, AlunoRA, P1, APS, Prog, N2)

VALUES (10, 100, 5, 8, 6, 5),

(10, 101, 2, Null, 7, 3),

(11, 100, Null, 8, 6, 8),

(11, 101, 6, 4, 7, 5),

(12, 103, 4, 8, Null, 8),

(13, 101, 5, 5, 7, 2),

(13, 104, 8, 9, 10, Null),

(13, 103, 8, 5, 10, 4)

Agora, crie uma função chamada **NotaCompleta** que recebe o CRT de uma disciplina e o RA de um aluno e retornar uma tabela contendo todos os campos de Nota e a N1 e NotaFinal calculadas pela própria função. A chamada da função **NotaCompleta** deve ser feita no seguinte formato:

SELECT \* FROM **NotaCompleta**(RA,CRT)

1. No caso acima, seria possível fazer uma coluna aprovação aparecer contendo ‘Aprovado’ se a nota final for maior ou igual a 5 e ‘Não aprovado’ se a nota for menor do que 5? Como?