UNIP – Universidade Paulista

Campus Tatuapé

**Linguagem de Programação para Banco de Dados**

Trabalho Semestral

ANDREICK GOMES MAIA - F044289

BEATRIZ ALMEIDA DE LIMA SOUZA - N459JB4

MAICOLN VINICIUS DOS S. SILVA - F0551I7

MATEUS CARVALHO DA SILVA - N508620

São Paulo - SP

# Etapa 1 – Modelagem do Banco de Dados

Figura Visão Geral do Banco UNIB\_DB

# 

Figura Tabela RESPONSÁVEL

# 

Figura Tabela CURSO

# 

Figura Tabela COORDENADOR

# 

Figura Tabela ALUNO

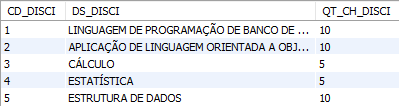


Figura Tabela DISCIPLINA

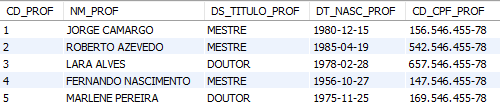


Figura Tabela PROFESSOR

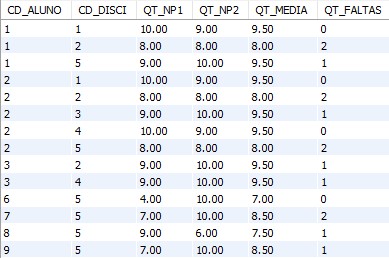


Figura Tabela ALUNO\_HAS\_DISCIPLINA

# Etapa 2 – Consultas Complexas

**Enunciado 1)** O departamento de recursos humanos precisa de um relatório de todos os alunos e os seus respectivos cursos. Crie uma consulta que exiba o nome do responsável, o nome do aluno, o CPF, o nome do curso, o tipo de curso, a carga horária e a quantidade de disciplinas do aluno.

**Resolução:**

select R.nm\_resp, A.nm\_aluno, A.cd\_cpf\_aluno, C.ds\_curso, C.ds\_tipo\_curso, C.qt\_ch\_curso, count(CD.cd\_curso) qt\_disciplinas

from responsavel R

join aluno A

on R.cd\_aluno = A.cd\_aluno

join curso C

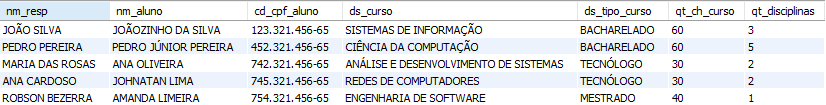
on A.cd\_curso = C.cd\_curso

join curso\_has\_disciplina CD

on C.cd\_curso = CD.cd\_curso

group by A.cd\_aluno;

**Resultado:**

****

**Enunciado 2)** O departamento de recursos humanos precisa de um relatório de todas as disciplinas com os professores que as lecionam e o curso a que pertencem. Crie uma consulta que exiba o nome do professor, o título do professor, o nome da disciplina, o nome do curso, o tipo de curso e o nome do coordenador. Exiba também as disciplinas que não tenham um professor ou não estejam em um curso ou que o curso não tenha um coordenador.

**Resolução:**

select P.nm\_prof, P.ds\_titulo\_prof, D.ds\_disci, Cu.ds\_curso, Cu.ds\_tipo\_curso, Co.nm\_coord

from coordenador Co

right join curso Cu

on Co.cd\_curso = Cu.cd\_curso

join curso\_has\_disciplina CD

on Cu.cd\_curso = CD.cd\_curso

right join disciplina D

on CD.cd\_disci = D.cd\_disci

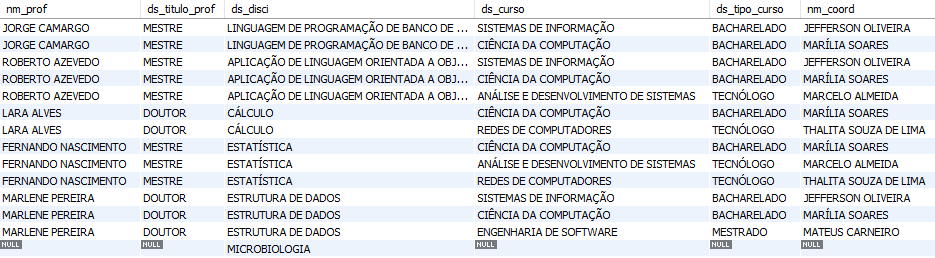
left join disciplina\_has\_professor DP

on D.cd\_disci = DP.cd\_disci

left join professor P

on DP.cd\_prof = P.cd\_prof;

**Resultado:**

****

**Enunciado 3)** Faça uma listagem de todos os alunos de Sistemas de Informação que foram aprovados (média maior ou igual a sete) em Estrutura de Dados.

**Resolução:**

select cd\_aluno, nm\_aluno

from aluno

where cd\_curso in (

select cd\_curso

from curso

where ds\_curso = 'sistemas de informação'

) and cd\_aluno in (

select cd\_aluno

from aluno\_has\_disciplina

where cd\_disci in (

select cd\_disci

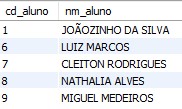
from disciplina

where ds\_disci = 'estrutura de dados'

) and qt\_media >= 7

);

**Resultado:**

****

# Etapa 3 – Functions, Triggers e Procedures

**Procedure 1)** Crie uma procedure consulta\_aluno, que liste, nome do aluno, data de nascimento, cpf, nome do responsável, nome do curso e o tipo do curso, para os alunos com nome ou curso informados por parâmetro.

**Resolução:**

create procedure consulta\_aluno (x varchar(45))

begin

select a.nm\_aluno as nome,

a.dt\_nasc\_aluno as nascimento,

a.cd\_cpf\_aluno as cpf,

r.nm\_resp as responsavel,

c.ds\_curso as curso,c.ds\_tipo\_curso as tipo

from aluno a

inner join responsavel r

on a.cd\_aluno = r.cd\_aluno

inner join curso c

on c.cd\_curso = a.cd\_curso

where a.nm\_aluno like concat('%', x , '%') or c.ds\_curso like concat('%', x , '%');

end

**Procedure 2)** Crie uma procedure cad\_nota\_aluno, que liste, nome do aluno, nome do curso e insira as notas na tabela aluno\_has\_disciplina com os dados informados por parâmetro.

**Resolução:**

create procedure cad\_nota\_aluno(p\_cd\_aluno int,

p\_cd\_disci int (3),

p\_qt\_np1 double(3,2),

p\_qt\_np2 double(3,2),

p\_qt\_media double(3,2),

p\_qt\_faltas int)

begin

select a.nm\_aluno as nome,

c.ds\_curso as curso

from aluno a

inner join curso c

on a.cd\_curso = c.cd\_curso

where a.cd\_aluno = p\_cd\_aluno;

insert into aluno\_has\_disciplina values

(p\_cd\_aluno, p\_cd\_disci, p\_qt\_np1, p\_qt\_np2, p\_qt\_media,p\_qt\_faltas);

end

**Function 1)** Crie uma função total\_alunos que receba como parâmetro (cod int) o código do curso, e retorne o total de alunos no curso.

**Resolução:**

create function total\_alunos (cod int)

returns int

reads sql data

deterministic

begin

declare total int default 0;

select count(cd\_aluno) into total

from aluno

where cd\_curso = cod;

return total;

end;

**Trigger 1)** Crie uma trigger total\_alunos que, após a inclusão de um aluno, atualize a tabela qt\_vagas somando um na quantidade

**Resolução:**

create trigger inclui\_aluno

after insert on aluno

for each row

begin

update qt\_vagas

set qtd = qtd + 1;

end;

**Trigger 2)** Crie uma trigger del\_aluno que, antes da exclusão de um aluno, exclua seu responsável, suas notas e faltas

**Resolução:**

create trigger del\_aluno

before delete on aluno

for each row

begin

delete from aluno\_has\_disciplina where cd\_aluno = old.cd\_aluno;

delete from responsavel where cd\_aluno = old.cd\_aluno;

end;