

Sistema de Auxílio à Secretaria do Departamento de Informática

INF1383

**Clara Szwarcman**

**Daniel Siles**

**Fernando Roisman**

**Lucas Borges**

**Guilherme Simas**

**Simon Travancas**

Índice

1. Introdução ................................................................3
2. Diagrama E-R.............................................................5
3. Descrição Entidades e Atributos.............................6
4. Esquema Lógico-Relacional...................................10
5. Restrições de Integridade Semântica...................11
6. Criação das Tabelas ...............................................12
7. Consultas em SQL.....................................................17
8. Consultas em Álgebra Relacional.........................20
9. Views .........................................................................21
10. Triggers.......................................................................25
11. Procedimentos..........................................................34
12. Funções .....................................................................35
13. Índices........................................................................37
14. Avaliação do Projeto ..............................................38
15. Introdução

O trabalho de administração e gerência dos cursos oferecidos pela Universidade não é trivial, e ferramentas que auxiliem neste processo são sempre bem vindas. Oferecer uma forma simples e expressiva de realizar consultas ao universo de informações disponíveis sobre os aspectos funcionais da vida acadêmica ao Departamento de Informática é o principal objetivo deste trabalho.

O sistema permitirá o acesso e operações a informações tais como: quais alunos dão monitoria no departamento e em quais cadeiras; quais alunos estagiam em quais laboratórios do departamento; qual professor orienta o projeto final do aluno e qual é o tema do projeto; qual professor orienta a tese do aluno e qual é o tema da tese e quais alunos são bolsistas e de que tipo.

Para a construção do banco de dados será necessário conhecer as seguintes informações:

Para cada aluno sua matrícula e seu nome. Se for de graduação, seu curso. Se for de pós-graduação serão necessárias mais informações pessoais, como os dados do passaporte, identidade, endereço e uma conta bancária, que poderá estar associada a mais de um aluno, assim como se o aluno é de mestrado ou doutorado. Para cada professor, seu nome e sua matrícula.

Um aluno pode ser monitor de até uma disciplina, enquanto uma disciplina pode possuir vários monitores. Sobre a disciplina é necessário conhecer seu código e nome. Também é preciso saber de qual turma o aluno foi monitor, a data de início e de fim da monitoria.

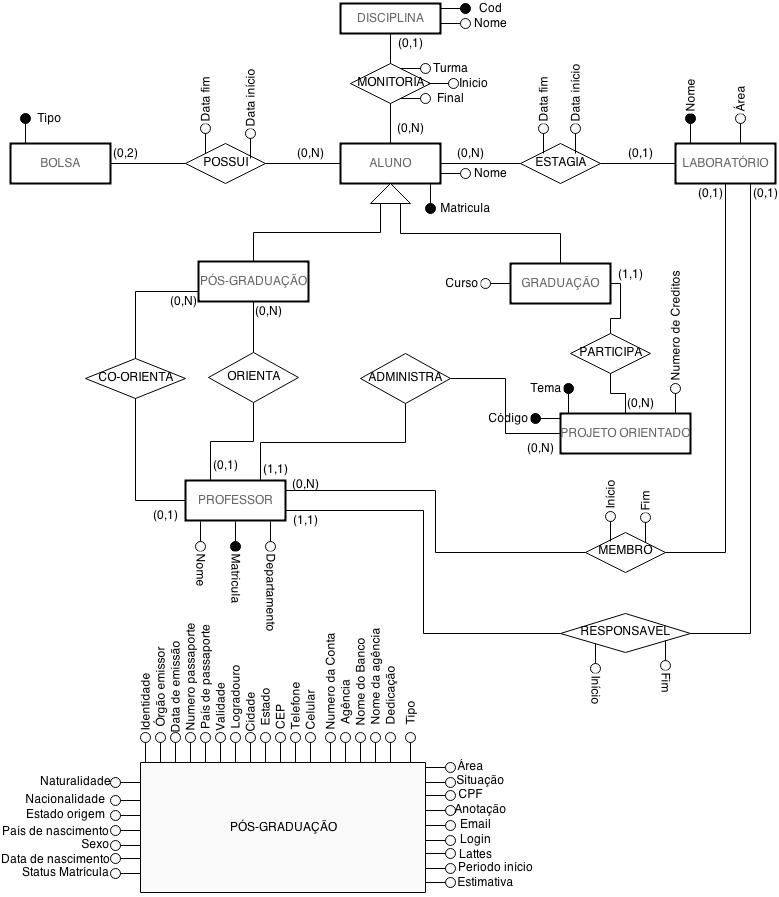
Um professor pode orientar e co-orientar vários alunos de pós-graduação. Um aluno de pós-graduação pode possuir até no máximo um orientador e um co-orientador.

Um professor também pode administrar vários projetos orientados, nos quais podem participar vários alunos de graduação. Um projeto possui um tema, número de créditos e um código. Entretanto, para um mesmo projeto o número de créditos pode variar para os alunos participantes. Também será conhecida a nota de cada aluno no projeto. Cada projeto é administrado por somente um professor. Um aluno de graduação pode participar de vários projetos.

Um aluno pode ser estagiário em até um laboratório. É necessário saber a data de início e de fim do estágio. Um laboratório possui, além de um nome e uma área, um professor responsável, e professores membros sem restrição de quantidade. Um professor só pode pertencer a um laboratório, seja como responsável ou membro.

Cada aluno pode possuir até duas bolsas simultaneamente, uma bolsa possui um tipo e pode estar associada a diversos alunos. Será conhecida a data de início do recebimento da bolsa e, se a bolsa não for mais vigente, a data de término.

1. Diagrama de Entidade-Relacionamento



1. Descrição Entidades e Atributos

**Entidades e atributos**

1. Aluno – Contém os alunos que se necessita cadastrar.

* Matrícula – número de 7 dígitos gerado que identifica o aluno
* Nome – texto que se refere ao nome do aluno (máximo de 50 caracteres)

1.1 Graduação – Contém os alunos de graduação cadastrados

* Curso –texto que se refere ao curso do aluno (máximo de 30 caracteres)

1.2 Pós-Graduação – Contém os alunos de pós-graduação cadastrados

* Tipo – texto referente a que tipo de pós-graduação o aluno cursa (máximo de 9 caracteres)
  + Pode ser “doutorado”,”mestrado”.
* Área – texto que se refere à área de estudo do aluno (máximo de 50 caracteres)
* Situação - texto que se refere a situação do aluno na pós-graduação (máximo de 10 caracteres)
  + Pode ser: “afastou-se”, “em curso”, “atrasado”, “formando”,”formado”.
* CPF – Combinação de 14 caracteres (seguindo o modelo: \*\*\*.\*\*\*.\*\*\*-\*\*) referente ao Cadastro de Pessoa Física
* Anotação – texto referente à alguma anotação, se desejado (máximo de 200 caracteres)
  + Campo de observações
* E-mail – texto referente ao e-mail do aluno (máximo de 30 caracteres)
  + Obrigatoriamente o da PUC (@inf.puc-rio.br)
* Login – texto gerado para acesso do aluno (máximo de 20 caracteres)
* Lattes – link referente ao currículo lattes do aluno (máximo de 50 caracteres)
  + Formato: “http://lattes.cnpq.br/xxxxxxxxxxxx”
* Período Início – número de 4 digitos na parte antes do separador decimal e 1 depois, referente ao período de início
  + “2014.1”
* Estimativa – número do período que é estimado para a conclusão
  + Mesmo formato de período início
* Status matrícula – texto referente a situação atual da matrícula (máximo de 12 caracteres)
  + Pode ser: “prorrogação”, “marcou banca”, “matriculado”, “reabrindo”, “afastou-se”, “trancamento”,”formado”
* Data Nascimento – Data referente ao nascimento do aluno (AAAA-MM-DD)
* Sexo – Caracter que indica o sexo do aluno (‘F’ para feminino e ‘H’ para masculino)
* País Nascimento – texto referente ao país em que o aluno nasceu (máximo de 20 caracteres)
  + Pode ser null
* Estado Origem – texto referente ao estado em que o aluno nasceu (máximo de 30 caracteres)
  + Pode ser null
* Nacionalidade – texto referente à nacionalidade do aluno (máximo de 30 caracteres)
* Naturalidade – texto referente à naturalidade do aluno (máximo de 30 caracteres)
* Identidade – número de até 10 dígitos referente ao número do Registro Geral
* Órgão Emissor – texto referente ao órgão que emitiu a identidade (máximo de 30 caracteres)
* Data de emissão – data referente à emissão da identidade (AAAA-MM-DD)
* Número Passaporte (se estrangeiro) – texto referente ao número do passaporte do aluno (máximo de 20 caracteres)
* País Passaporte – texto referente ao país do passaporte do aluno (máximo de 30 caracteres)
* Validade Passaporte – data referente a validade do passaporte (AAAA-MM-DD)
* Logradouro – texto referente ao endereço residencial (máximo de 30 caracteres)
* Cidade – texto referente a cidade em que o aluno mora (máximo de 30 caracteres)
* Estado – texto referente ao estado em que o aluno mora (máximo de 30 caracteres)
* CEP – número de 8 dígitos referente ao Código de Endereçamento Postal
* Telefone – número de até 11 dígitos referente ao telefone residencial do aluno
* Celular – número de até 11 dígitos referente ao celular do aluno
* Dedicação - texto que se refere ao tipo de dedicação do aluno (máximo de 9 caracteres)
  + Pode ser: “parcial”, “exclusiva”.

1. Disciplina – Contém as disciplinas do departamento de informática
   * Código – texto gerado para identificar uma disciplina (3 letras maiúsculas seguidas de 4 números)
   * Nome – texto referente ao nome da disciplina (máximo de 50 caracteres)
2. Professor – Contém os professores que se necessita cadastrar
   * Matrícula – número de 7 dígitos que identifica um professor
   * Nome – texto referente ao nome do professor (máximo de 50 caracteres)
   * Departamento – texto referente ao departamento que o professor pertence
3. Laboratório – Contém os laboratórios do departamento de informática
   * Nome – texto referente ao nome do laboratório (máximo de 50 caracteres)
   * Área – texto referente a área de atuação do laboratório (máximo de 50 caracteres)
4. Bolsa – Contém as bolsas cadastradas na faculdade
   * Tipo/Órgão – texto referente ao tipo da bolsa (máximo de 50 caracteres)
5. Conta Banco – Contém as contas de banco cadastradas
   * Nº do banco – texto referente ao número do banco do aluno (máximo de 4 caracteres)
   * Nº da agência – número referente ao número da agência do aluno (máximo de 8 algarismos)
   * Nº da conta – número referente ao número da conta do aluno (máximo de 4 algarismos)
   * Nome da Agência – texto referente ao nome da agência do aluno (máximo de 30 caracteres)
6. Projeto Orientado – Contém os Projetos Orientados cursados pelos alunos do departamento
   * Código – texto gerado que identifica um projeto orientado (3 letras maiúsculas seguidas de 4 números)
   * Tema – texto referente ao tema do projeto orientado (máximo de 50 caracteres)
   * Número de Créditos – número referente à quantidade de créditos do projeto orientado (máximo de 2 algarismos)
   * Período – número do período que foi realizado o projeto
     + Formato: 2014.1
   * Nota – número da nota obtida no projeto (máximo de 2 algarismos antes da vírgula e 1 depois)

**Relacionamentos**

1. Monitoria – Relação entre um Aluno e uma Disciplina, na qual o aluno é monitor da disciplina. O aluno só pode ser monitor de uma disciplina, a disciplina pode ter vários monitores.
   * Turma – texto referente a qual turma da disciplina o aluno é monitor
   * Data Início – data referente ao início da monitoria
   * Data Fim – data referente ao término da monitoria, se já tiver ocorrido o término
2. Possui – Relação entre um Aluno e uma Bolsa, na qual o aluno possui uma bolsa. Um aluno pode possuir até 2 bolsas e cada bolsa pode estar atribuída a diversos alunos.
   * Data Início – data referente ao início do recebimento da bolsa
   * Data Fim – data referente ao término do recebimento da bolsa, se já tiver ocorrido o término
3. Estágio – Relação entre um aluno e um laboratório, na qual um aluno estagia no laboratório. Um laboratório pode ter vários estagiários, mas um aluno só estagia em até um laboratório.
   * Data Início – data referente ao início do estágio pelo aluno
   * Data Fim – data referente ao término do estágio pelo aluno, se já tiver ocorrido o término
4. Membro – Relação entre um Professor e um Laboratório, na qual o professor é membro do laboratório. Um laboratório pode ter vários membros, e um professor só pode ser membro de até um laboratório.
   * Data Início – data referente à quando o professor passou a ser membro do laboratório
   * Data Fim – data referente à quando o professor deixou de ser membro do laboratório, se já tiver deixado de ser membro
5. Responsável - Relação entre um Professor e um Laboratório, na qual o professor é responsável pelo laboratório. Um professor pode ser responsável por até um laboratório, e um laboratório contém exatamente um responsável.
   * Data Início – data referente à quando o professor passou a ser responsável pelo laboratório
   * Data Fim – data referente à quando o professor deixou de ser responsável pelo laboratório, se já tiver deixado de ser responsável
6. Participa – Relação entre um Aluno e um Projeto Orientado na qual o aluno participa do projeto. O projeto tem um aluno participante, e o aluno pode participar de vários projetos.
7. Orienta Mestrando – Relação entre um Aluno de Mestrado e um Professor, na qual o professor orienta o aluno. Um mestrando pode ter até 1 orientador, e um professor pode orientar vários mestrandos.
8. Orienta Mestrando – Relação entre um Aluno de Doutorado e um Professor, na qual o professor orienta o aluno. Um doutorando deve ter 1 orientador, e um professor pode orientar vários doutorandos.
9. Co-orienta - Relação entre um Aluno de Doutorado e um Professor, na qual o professor co-orienta o aluno. Um doutorando pode ter até 1 co-orientador, e um professor pode co-orientar vários doutorandos.
10. Administra – Relação entre um Professor e um Projeto Orientado, na qual o professor administra o projeto. Um projeto tem somente um professor que o administra, e um professor pode administrar vários projetos.
11. Vinculado – Relação entre uma Conta de Banco e um aluno de Pós-Graduação, na qual o aluno está vinculado a uma conta. Cada aluno possui exatamente uma conta de banco, e cada conta de banco está vinculada a pelo menos um aluno.
12. Esquema Lógico - Relacional

**Tabelas**

* ALUNO (Matricula ,Nome)
* GRAD (Matricula , Curso)
  + Matricula referencia ALUNO

* POSGRAD (*Matricula*, Tipo, Area, Situacao, CPF, Anotacao, E-mail, Login, Lattes, PeriodoInicio, Estimativa, StatusMatricula, DataNascimento, Sexo, PaisNascimento, EstadoOrigem, Nacionalidade, Naturalidade, Identidade, OrgaoEmissor, DataEmissao, NumPassaporte, PaisPassaporte, ValidadePassaporte, Logradouro, Cidade, Estado, CEP, Telefone, Celular, NumBanco, NumConta, NumAgencia, NomeAgencia, Dedicação, *Orientador*, *Co-orientador*).
  + Matricula referencia ALUNO
  + Orientador referencia PROFESSOR (Relação Orienta). (Pode ser nulo)
  + Co-orientador referencia PROFESSOR (Relação Co-orienta). (Pode ser nulo)
* PROFESSOR (Matricula, Nome, Departamento, *NomeLab*, DataIniMembro, DataFimMembro).
  + NomeLab referencia LABORATORIO (Relação Membro). (Pode ser nulo)
* DISCIPLINA (Codigo, Nome).
* LABORATORIO (Nome, Area, *Responsavel*).
  + Responsavel referencia PROFESSOR. (Relação responsável)
* BOLSA (Tipo, *Aluno*, DataIni, DataFim).
  + Aluno referencia ALUNO (Relação Possui). (Pode ser Nulo)
* MONITORIA (*Aluno*, *CodDisc*, DataIni, DataFim)
  + Aluno referencia ALUNO.
  + CodDisc referencia Disciplina
* ESTAGIOLAB (*Aluno*, *Laboratorio*, DataIni, DataFim)
  + Aluno referencia ALUNO.
  + Laboratorio referencia LABORATORIO
* PROJETOORIENTADO (Codigo, *Aluno*, Tema, NumCred, Nota, Periodo, *MatriculaProf*).
  + Aluno referencia GRAD.
  + MatriculaProf referencia PROFESSOR (Relação Administra).

1. Restrições de Integridade Semântica

Foram definidas as seguintes Regras de Negócio:

* Um aluno de pós-graduação cursando um doutorado tem obrigatoriamente um orientador.
* Um aluno de pós-graduação cursando um mestrado pode não ter um orientador durante a primeira metade do programa (o mestrado dura 4 períodos).
* Um aluno não pode ser responsável por mais de uma monitoria ao mesmo tempo. Não deve haver interseção entre o fim de uma monitoria e o início de outra.
* Um aluno não pode estagiar em mais de um laboratório ao mesmo tempo. Não deve haver interseção entre o fim de um estágio e o início de outro.
* Um doutorando possui dedicação exclusiva ao curso obrigatoriamente.

1. Criação das Tabelas

A seguir estão os códigos em SQL referentes à criação das tabelas especificadas na seção IV.

|  |
| --- |
| CREATE TABLE aluno |
| ( |
| matricula numeric(7) NOT NULL, |
|  |
| Constraint pk\_aluno Primary Key (matricula), |
|  |
| nome varchar(50) NOT NULL |
|  |
| ); |
|  |
| CREATE TABLE grad |
| ( |
| matricula numeric(7) NOT NULL, |
|  |
| Constraint fk\_graduacao Foreign Key (matricula) |
| References aluno (matricula), |
|  |
| curso varchar(30) NOT NULL |
|  |
| ); |
|  |
| CREATE TABLE posgrad |
| ( |
| matricula numeric(7) NOT NULL, |
|  |
| Constraint fk\_posgrad\_matricula Foreign Key (matricula) |
| References aluno (matricula), |
| CONSTRAINT pk\_posgrad Primary Key (matricula), |
|  |
| tipo varchar(9) NOT NULL, |
|  |
| Constraint CKC\_tipo\_posgrad |
| Check (tipo IN ('doutorado','mestrado')), |
|  |
| area varchar(50) NOT NULL, |
| situacao varchar(10) NOT NULL, |
|  |
| Constraint CKC\_situacao\_posgrad |
| Check (situacao IN ('afastou-se','em curso','atrasado','formando',’formado’)), |
|  |
| CPF char(14) NOT NULL, |
|  |
| CONSTRAINT CKC\_cpf CHECK (cpf like ('\_\_\_.\_\_\_.\_\_\_-\_\_')), |
| CONSTRAINT UNI\_cpf UNIQUE (cpf), |
|  |
| anotacao varchar(200) , |
| email varchar(30) NOT NULL, |
|  |
| Constraint CKC\_email |
| CHECK (email like ('%@inf.puc-rio.br')), |
|  |
| login varchar(20) NOT NULL, |
|  |
| CONSTRAINT UNI\_login UNIQUE (login), |
|  |
| lattes varchar(50) NOT NULL, |
|  |
| CONSTRAINT CKC\_lattes CHECK (lattes like ('http://lattes.cnpq.br/%')), |
| CONSTRAINT UNI\_lattes UNIQUE (lattes), |
|  |
| periodoinicio numeric(5,1) NOT NULL, |
| estimativa numeric(5,1) NOT NULL, |
| status\_matricula varchar(12) NOT NULL, |
|  |
| Constraint CKC\_status\_posgrad |
| Check (status\_matricula IN ('prorrogação','marcou banca','matriculado', |
| 'reabrindo','afastou-se','trancamento',’formado’)), |
|  |
| data\_nascimento date NOT NULL, |
| sexo char(1) NOT NULL, |
|  |
| Constraint CKC\_sexo\_posgrad |
| Check (sexo IN ('F','H')), |
|  |
| pais\_nascimento varchar(20), |
| estado\_origem varchar(30), |
| nacionalidade varchar(30) NOT NULL, |
| naturalidade varchar(30) NOT NULL, |
| id varchar(10), |
| orgao\_emissor varchar(30), |
| data\_emissao date, |
| num\_passaporte varchar(20), |
| pais\_passaporte varchar(30), |
| validade\_passaporte date, |
| logradouro varchar(30) NOT NULL, |
| cidade varchar(30) NOT NULL, |
| estado varchar(30) NOT NULL, |
| cep numeric(8) NOT NULL, |
| telefone numeric(11), |
| celular numeric(11), |
| num\_banco char(4) NOT NULL, |
| num\_conta numeric(8) NOT NULL, |
| num\_agencia numeric(4) NOT NULL, |
| nome\_agencia varchar(30) NOT NULL, |
| dedicacao varchar(9) NOT NULL, |
|  |
| Constraint CKC\_dedicacao\_posgrad |
| Check (dedicacao IN ('parcial','exclusiva')), |
|  |
| orientador numeric(7), |
|  |
| Constraint fk\_posgrad\_orientador Foreign Key (orientador) |
| References professor (matricula), |
|  |
| coorientador numeric(7), |
|  |
| Constraint fk\_posgrad\_coorientador Foreign Key (coorientador) |
| References professor (matricula) |
|  |
| ); |
|  |
|  |
| CREATE TABLE professor |
| ( |
| matricula numeric(7) NOT NULL, |
|  |
| Constraint pk\_professor Primary Key (matricula), |
|  |
| nome varchar(50) NOT NULL, |
| nome\_lab varchar(50), |
|  |
|  |
| data\_ini\_membro date , |
| data\_fim\_membro date |
|  |
| ); |
| ALTER TABLE professor ADD Constraint fk\_professor\_nome\_lab Foreign Key (nome\_lab) |
| References laboratorio (nome); |
|  |
|  |
| CREATE TABLE disciplina |
| ( |
| codigo char(7) NOT NULL, |
|  |
| Constraint pk\_disciplina Primary Key (codigo), |
|  |
| nome varchar(50) NOT NULL |
|  |
| ); |
|  |
| CREATE TABLE laboratorio |
| ( |
| nome varchar(50) NOT NULL, |
|  |
| Constraint pk\_laboratorio Primary Key (nome), |
|  |
| area varchar(50) NOT NULL, |
| responsavel numeric(7) NOT NULL, |
|  |
| Constraint fk\_laboratorio\_responsavel Foreign Key (responsavel) |
| References professor (matricula) |
|  |
| ); |
|  |
| CREATE TABLE bolsa |
| ( |
| tipo varchar(50) NOT NULL, |
|  |
| aluno numeric(7) NOT NULL, |
|  |
| Constraint fk\_bolsa\_aluno Foreign Key (aluno) |
| References aluno (matricula), |
|  |
| data\_ini date NOT NULL, |
|  |
| Constraint pk\_bolsa Primary Key (aluno,tipo,data\_ini), |
|  |
| data\_fim date |
|  |
| ); |
|  |
| CREATE TABLE monitoria |
| ( |
| aluno numeric(7) NOT NULL, |
|  |
| Constraint fk\_monitoria\_aluno Foreign Key (aluno) |
| References aluno (matricula), |
|  |
| codigo char(7) NOT NULL, |
|  |
| Constraint fk\_monitoria\_codigo Foreign Key (codigo) |
| References disciplina (codigo), |
|  |
| data\_ini date NOT NULL, |
|  |
| Constraint pk\_monitoria Primary Key (codigo,aluno,data\_ini), |
|  |
| data\_fim date |
|  |
| ); |
|  |
| CREATE TABLE estagiolab |
| ( |
| aluno numeric(7) NOT NULL, |
|  |
| Constraint fk\_estagiolab\_aluno Foreign Key (aluno) |
| References aluno (matricula), |
|  |
| laboratorio varchar(50) NOT NULL, |
|  |
| Constraint fk\_estagiolab\_laboratorio Foreign Key (laboratorio) |
| References laboratorio (nome), |
|  |
| data\_ini date NOT NULL, |
|  |
| Constraint pk\_estagiolab Primary Key (laboratorio,aluno,data\_ini), |
|  |
| data\_fim date |
| ); |
|  |
| CREATE TABLE projetoorientado |
| ( |
| aluno numeric(7) NOT NULL, |
|  |
| Constraint fk\_projeto\_orientado\_aluno Foreign Key (aluno) |
| References aluno (matricula), |
|  |
| codigo char(7) NOT NULL, |
|  |
| Constraint pk\_projeto\_orientado Primary Key (codigo,aluno), |
|  |
| tema varchar(50) NOT NULL, |
| numcred numeric(2) NOT NULL, |
| nota numeric(3,1) NOT NULL, |
| periodo numeric(5,1) NOT NULL, |
| matricula\_prof numeric(7) NOT NULL, |
|  |
| Constraint fk\_projeto\_orientado\_matricula\_prof Foreign Key (matricula\_prof) |
| References professor (matricula)  ); |
|  |

1. Consultas em SQL

**1 - Todos os atributos (\*), Subconsulta aninhada, IN, NOT LIKE '%', ORDER BY, IS NULL**

Quais são os alunos de graduação que dão monitoria e não fazem Engenharia?

**SELECT** \*

**FROM** Aluno

**WHERE** A.Matricula IN ( **SELECT** G.matricula

**FROM** Graduacao as G, Monitoria as M

**WHERE** G.matricula = M.aluno

**AND** G.curso NOT LIKE '%Engenharia%'

**AND** M.datafim IS NULL)

**ORDER BY** Nome

**2 - LEFT OUTER JOIN**

Listar todos os monitores, e se estagiarem em algum laboratório, o nome do laboratório.

**SELECT** M.Matricula, M.Nome, E.Nome

**FROM** MONITORIA as M left outer join ESTAGIOLAB as E on M.Aluno = E.Aluno

**3 - SUM , INNER JOIN**

Qual a quantidade de créditos administrados pelo professor "Sergio Lifschitz" na disciplina Projeto Orientado?

**SELECT** SUM NumCred

**FROM** PROJETOORIENTADO INNER JOIN PROFESSOR ON  PROJETOORIENTADO.MatriculaProf = PROFESSOR.Matricula

**WHERE** PROFESSOR.nome = 'Sergio Lifschitz'

**4- COUNT**

Quantos alunos de Pos-Graduação se matricularam em 2013?

**SELECT** COUNT (\*)

**FROM** POSGRAD

**WHERE** POSGRAD.PeriodoInicio like '2013.?'

**5 - INTERSECT**

Quais professores são orientadores e co-orientadores?

**SELECT** Nome

**FROM** PROFESSOR

**WHERE** Matricula IN

            (**SELECT** P.Matricula

**FROM** PROFESSOR as P, POSGRAD as PG

**WHERE** PG.Co-orientador=P.Matricula)

**INTERSECT**

            (**SELECT** P.Matricula

**FROM** PROFESSOR as P, POSGRAD as PG

**WHERE** PG.Orientador=P.Matricula)

**6 - EXCEPT**

Quais os atuais monitores de Programação I?

**SELECT** Aluno

**FROM** MONITORIA

**WHERE** CodDiscp = ‘INF1005’

**EXCEPT**(

**SELECT** Aluno

**FROM** MONITORIA

**WHERE** DataFim IS NULL

            )

**7 - UNION**

Quais alunos são monitores, estagiários, ou ambos?

**SELECT** Aluno

**FROM** MONITORIA

**WHERE** DataFim is NULL

**UNION** (

**SELECT** Aluno

**FROM** ESTAGIOLAB

**WHERE** DataFim is NULL

)

**8 - SOME**

Quais laboratórios tem 3 ou 7 membros?

**SELECT** NomeLab

**FROM** ( **SELECT** NomeLab, COUNT(Matrícula) AS NumMembros

**FROM** Professor GROUP BY NomeLab ) AS A

**WHERE** NumMembros = SOME (3,7)

**9 - ALL**

Quais laboratórios nao tem 2 nem 5 membros?

**SELECT** NomeLab

**FROM** ( **SELECT** NomeLab, COUNT(Matrícula) AS NumMembros

**FROM** Professor GROUP BY NomeLab ) AS A

**WHERE** NumMembros != ALL (2,5)

**10 - EXISTS**

Quais professores deram um projeto orientado no período de 2014.1?

**SELECT** P.matricula, P.nome

**FROM** professor as P

**WHERE** EXISTS ( **SELECT** \*

**FROM** projetoorientado as PJO

**WHERE** PJO.matriculaprof = P.matricula

**AND** PJO.periodo=2014.1 )

**11 - '????'**

Quais os alunos que tem o primeiro nome com 5 letras?

**SELECT** A.matricula, A.nome

**FROM** Aluno as A

**WHERE** A.nome LIKE ‘????? %’

**12 - Having**

Quantos alunos deram mais de uma monitoria?

**SELECT** aluno.matricula, aluno.nome, count(\*)

**FROM** aluno, monitoria

**WHERE** aluno.matricula=monitoria.aluno

**AND EXISTS** (SELECT \*

FROM monitoria

WHERE monitoria.aluno=aluno.matricula

AND monitoria.data\_fim IS NULL)

**GROUP BY** aluno.matricula, aluno.nome

**HAVING** count(\*)>1

**13 – Unique**

Quantos alunos deram mais de uma monitoria?

**SELECT** DISTINCT aluno.matricula,aluno.nome

**FROM** aluno

**WHERE** **NOT UNIQUE** (

**SELECT** (\*)

**FROM** monitoria

**WHERE** aluno.matricula=monitoria.aluno

)

1. Consultas em Álgebra Relacional

**1 – “÷” (divisão), “|><|” (junção natural)**

Listar todos os alunos que foram ou são monitores de prog 1 e prog 2.

DISCIPLINAS ← π <Codigo> (σ <Nome = Prog I II Nome = Prog II (DISCIPLINA))

ALUNOS ← MONITORIA ÷ DISCIPLINAS

RESP ← π <Matricula,Nome > (Alunos |><| (Aluno=Matricula) (ALUNO))

**2 - “U” (união), “×” (produto cartesiano)**

Quais alunos são monitores, estagiários ou ambos?

MONITORES ← π <Aluno> (σ <DataFim = null> (MONITORIA))

ESTAGIARIOS ← π <Aluno> (σ <DataFim = null> (ESTAGIOLAB))

AMBOS ← MONITORES ∪ ESTAGIARIOS

TEMP1 ← σ<Aluno = Matricula> (AMBOS X ALUNO)

RESP ← π <Matricula, Nome > (TEMP1)

**3 -** **“G” (aggregate functions)**

Qual a quantidade de créditos administrados pelo professor "Sergio Lifschitz" na disciplina Projeto Orientado?

TEMP1 ← MatriculaProf G sum (NumCred) (PROJETOORIENTADO)

TEMP2 ← TEMP1 |><|(MatriculaProf = Matricula) PROFESSOR

TEMP3 ← σ <Nome = ‘Sergio Lifschitz’> (TEMP2)

RESP ← π <Nome, NumCred> (TEMP3)

**4 - “∩” (interseção)**

Quais alunos de pós-graduação já deram monitoria?

MONITORES ← π <Aluno> (MONITORIA)

POS ← π <Matricula> (POSGRAD)

TEMP1 ← MONITORES **∩** POS

TEMP2 ← σ <Aluno = Matricula> (TEMP1 X ALUNO)

TEMP3 ← π <Matricula, Nome> (TEMP2)

**5 -** **“=|><|**” **(junção externa à esquerda)**

Listar todos os alunos dizendo se eles são monitores, de que disciplina, a data do início, e data do fim da monitoria

R1 ← σ <Codigo = CodDisc> (DISCIPLINA X MONITORIA)

R2 ← σ <Nome = NomeDisc> (R1)

R3 ← Aluno =|><| <Matricula = Aluno> R2

Resp ← π <Nome, NomeDisc, DataIni, DataFim> (R3)

**6 -** **Projeção generalizada**

R1← G sum <numcred>(PROJETOORIENTADO)

R2← RENAME<R2(numcredtotal)>(R1)

R3← <matricula\_prof> G sum <numcred>(PROJETOORIENTADO)

R4← R1 X R2

RF← π <matricula\_prof,numcred/numcredtotal>(R4)

1. Views

**Obtem todos os alunos de graduação e seus respectivos nomes:**

|  |
| --- |
| create view alunos\_pos as |
| (select \*  from aluno natural join grad) |
|  |

**Obtem todos os alunos de pos graduação e seus respectivos nomes:**

|  |
| --- |
| create view alunos\_pos as |
| (select \* |
| from aluno natural join posgrad)  **Obtem todos os alunos de doutorado e seus respectivos nomes:** |
|  |
| create view alunos\_doutorado as |
| (select \* |
| from aluno natural join posgrad |
| where tipo = 'doutorado') |
| **Obtem todos os alunos de mestrado e seus respectivos nomes:** |
| create view alunos\_mestrado as |
| (select \* |
| from aluno natural join posgrad |
| where tipo = 'mestrado')  **Obtem todos os monitores e seus respectivos nomes:** |
|  |
| create view monitores as |
| (select \* |
| from monitoria, aluno |
| where aluno = matricula)  **Obtem todos os monitores da graduação que não fazem engenharia:** |
|  |
|  |
| create view monitores\_grad\_nao\_eng as |
| ( |
| SELECT \* |
| FROM aluno |
| WHERE matricula IN ( SELECT G.matricula |
| FROM grad as G, monitoria as M |
| WHERE G.matricula = M.aluno |
| AND G.curso NOT LIKE '%Engenharia%' |
| AND M.data\_fim IS NULL) |
| ORDER BY nome |
| where aluno = matricula)  **Obtem todos os monitores que também estagiam:** |
|  |
|  |
| create view monitor\_e\_estagiario as |
| ( |
| SELECT M.matricula, M.nome, E.laboratorio |
| FROM monitores as M left outer join estagiolab as E on M.aluno = E.aluno |
| )  **Obtem todos os pós graduados que iniciaram em 2013:** |
|  |
| create view posgrad\_2013 as |
| ( |
| SELECT COUNT (\*) |
| FROM posgrad |
| WHERE PeriodoInicio like '2013.?' |
| )  **Obtem a quantidade de créditos de projeto orientado dados pelo professor Sergio Lifschitz:** |
|  |
| create view cred\_sergio as |
| ( |
| SELECT SUM(numcred) |
| FROM projetoorientado INNER JOIN professor ON projetoorientado.matriculaprof = professor.matricula |
| WHERE professor.nome = 'Sergio Lifschitz' |
| )  **Obtem todos os professores que são orientadores e coorientadores:** |
|  |
| create view orient\_e\_coorient as |
| ( |
| SELECT nome |
| FROM professor |
| WHERE matricula IN |
| ( |
| (SELECT P.matricula |
| FROM professor as P, posgrad as PG |
| WHERE PG.coorientador=P.matricula) |
| INTERSECT |
| (SELECT P.matricula |
| FROM professor as P, posgrad as PG |
| WHERE PG.orientador=P.matricula) |
| ) |
| )  **Obtem todos os monitores de programação I:** |
|  |
| create view monitores\_progI as |
| ( |
| SELECT aluno |
| FROM monitoria |
| WHERE codigo = 'INF1005' |
| EXCEPT( |
| SELECT aluno |
| FROM monitoria |
| WHERE data\_fim IS NOT NULL |
| ) |
| )  **Obtem todos os alunos que são monitores e/ou estagiam:** |
|  |
| create view monitor\_eou\_estagiario as |
| ( |
| SELECT aluno |
| FROM monitoria |
| UNION ( |
| SELECT aluno |
| FROM estagiolab |
| ) |
| ) |
|  |
| create view projeto2014 as |
| ( |
| SELECT P.matricula, P.nome |
|  |
| FROM professor as P |
|  |
| WHERE EXISTS ( SELECT \* |
|  |
| FROM projetoorientado as PJO |
|  |
| WHERE PJO.matricula\_prof = P.matricula |
|  |
| AND PJO.periodo=2014.1 ) |
| ) |

1. Triggers

**Verifica que um aluno de pós graduação de doutorado possui um orientador:**

CREATE OR REPLACE FUNCTION tem\_orientador()

RETURNS TRIGGER AS $tg\_tem\_orientador$

BEGIN

IF new.tipo='doutorado' AND new.orientador is NULL THEN

RAISE EXCEPTION 'Doutorando % deve obrigatoriamente possuir um orientador.', NEW.matricula;

RETURN NULL;

END IF;

RETURN NEW;

END;

$tg\_tem\_orientador$ LANGUAGE plpgsql;

CREATE TRIGGER tg\_tem\_orientador BEFORE INSERT OR UPDATE ON posgrad

FOR EACH ROW EXECUTE PROCEDURE tem\_orientador();

**Verifica um aluno de pós graduação de doutorado possui dedicação exclusiva:**

CREATE OR REPLACE FUNCTION chk\_dedicacao()

RETURNS TRIGGER AS $tg\_chk\_dedicacao$

BEGIN

IF new.tipo='doutorado' AND new.dedicacao!='exclusiva' THEN

RAISE EXCEPTION 'Doutorando % deve obrigatoriamente possuir dedicacao exclusiva.', NEW.matricula;

RETURN NULL;

END IF;

RETURN NEW;

END;

$tg\_chk\_dedicacao$ LANGUAGE plpgsql;

CREATE TRIGGER tg\_chk\_dedicacao BEFORE INSERT OR UPDATE ON posgrad

FOR EACH ROW EXECUTE PROCEDURE chk\_dedicacao();

**Verifica que ao inserir uma monitoria, o aluno não possui outra em andamento:**

CREATE OR REPLACE FUNCTION chk\_monitoria\_insert()

RETURNS TRIGGER AS $tg\_chk\_monitoria\_insert$

DECLARE

aluno monitoria.aluno%TYPE;

BEGIN

SELECT monitoria.aluno into aluno

FROM monitoria

WHERE new.aluno = monitoria.aluno

AND (monitoria.data\_fim>new.data\_ini

OR monitoria.data\_fim is NULL);

IF aluno IS NOT NULL THEN

RAISE EXCEPTION 'Aluno % já possui/possuia uma monitoria em andamento.', new.aluno;

RETURN NULL;

END IF;

RETURN NEW;

END;

$tg\_chk\_monitoria\_insert$ LANGUAGE plpgsql;

CREATE TRIGGER tg\_chk\_monitoria\_insert BEFORE INSERT ON monitoria

FOR EACH ROW EXECUTE PROCEDURE chk\_monitoria\_insert();

**Verifica que ao atualizar uma monitoria, não aconteça de haver duas monitorias em um mesmo período de tempo:**

CREATE OR REPLACE FUNCTION chk\_monitoria\_update()

RETURNS TRIGGER AS $tg\_chk\_monitoria\_update$

DECLARE

aluno monitoria.aluno%TYPE;

BEGIN

SELECT monitoria.aluno into aluno

FROM monitoria

WHERE new.aluno = monitoria.aluno

AND monitoria.data\_fim>new.data\_ini

AND monitoria.data\_ini!=new.data\_ini;

IF aluno IS NOT NULL THEN

RAISE EXCEPTION 'Aluno % já possui/possuia uma monitoria em andamento.', new.aluno;

RETURN NULL;

END IF;

RETURN NEW;

END;

$tg\_chk\_monitoria\_update$ LANGUAGE plpgsql;

CREATE TRIGGER tg\_chk\_monitoria\_update BEFORE UPDATE ON monitoria

FOR EACH ROW EXECUTE PROCEDURE chk\_monitoria\_update();

**Verifica que ao inserir um estágio, o aluno não possui outro em andamento:**

CREATE OR REPLACE FUNCTION chk\_estagiolab\_insert()

RETURNS TRIGGER AS $tg\_chk\_estagiolab\_insert$

DECLARE

aluno estagiolab.aluno%TYPE;

BEGIN

SELECT estagiolab.aluno into aluno

FROM estagiolab

WHERE new.aluno=estagiolab.aluno

AND (estagiolab.data\_fim>new.data\_ini

OR estagiolab.data\_fim is NULL);

IF aluno IS NOT NULL THEN

RAISE EXCEPTION 'Aluno % já possui/possuia um estagio em laboratorio em andamento.', new.aluno;

RETURN NULL;

END IF;

RETURN NEW;

END;

$tg\_chk\_estagiolab\_insert$ LANGUAGE plpgsql;

CREATE TRIGGER tg\_chk\_estagiolab\_insert BEFORE INSERT ON estagiolab

FOR EACH ROW EXECUTE PROCEDURE chk\_estagiolab\_insert();

**Verifica que ao atualizar um estágio, , não aconteça de haver dois estágios em um mesmo período de tempo:**

CREATE OR REPLACE FUNCTION chk\_estagiolab\_update()

RETURNS TRIGGER AS $tg\_chk\_estagiolab\_update$

DECLARE

aluno estagiolab.aluno%TYPE;

BEGIN

SELECT estagiolab.aluno into aluno

FROM estagiolab

WHERE new.aluno=estagiolab.aluno

AND estagiolab.data\_fim>new.data\_ini

AND estagiolab.data\_ini!=new.data\_ini;

IF aluno IS NOT NULL THEN

RAISE EXCEPTION 'Aluno % já possui/possuia um estagio em laboratorio em andamento.', new.aluno;

RETURN NULL;

END IF;

RETURN NEW;

END;

$tg\_chk\_estagiolab\_update$ LANGUAGE plpgsql;

CREATE TRIGGER tg\_chk\_estagiolab\_update BEFORE UPDATE ON estagiolab

FOR EACH ROW EXECUTE PROCEDURE chk\_estagiolab\_update();

**Verifica que um aluno de pós-graduação brasileiro possui um número de identidade ,um órgão emissor e uma data de emissão:**

CREATE OR REPLACE FUNCTION chk\_brasileiro()

RETURNS TRIGGER AS $tg\_chk\_brasileiro$

DECLARE

nacionalidade posgrad.nacionalidade%TYPE;

BEGIN

SELECT posgrad.nacionalidade INTO nacionalidade

FROM posgrad

WHERE new.matricula=posgrad.matricula;

IF lower(new.nacionalidade) like 'brasileir\_' OR lower(nacionalidade) like 'brasileir\_' THEN

IF new.id IS NULL THEN

RAISE EXCEPTION 'Aluno % brasileiro precisa de um número de id.', new.matricula;

RETURN NULL;

END IF;

IF new.orgao\_emissor IS NULL THEN

RAISE EXCEPTION 'Aluno % brasileiro precisa de um órgão emissor.', new.matricula;

RETURN NULL;

END IF;

IF new.data\_emissao IS NULL THEN

RAISE EXCEPTION 'Aluno % brasileiro precisa de uma data de emissão.', new.matricula;

RETURN NULL;

END IF;

END IF;

RETURN NEW;

END;

$tg\_chk\_brasileiro$ LANGUAGE plpgsql;

CREATE TRIGGER tg\_chk\_brasileiro BEFORE INSERT OR UPDATE ON posgrad

FOR EACH ROW EXECUTE PROCEDURE chk\_brasileiro();

**Verifica que um aluno de pós-graduação estrangeiro possui um número de passaporte ,um país do passaporte e uma data de validade do passaporte:**

CREATE OR REPLACE FUNCTION chk\_estrangeiro()

RETURNS TRIGGER AS $tg\_chk\_estrangeiro$

BEGIN

IF lower(new.nacionalidade) not like 'brasileir\_' THEN

IF new.num\_passaporte IS NULL THEN

RAISE EXCEPTION 'Aluno % estrangeiro precisa de um número de passaporte.', new.matricula;

RETURN NULL;

END IF;

IF new.pais\_passaporte IS NULL THEN

RAISE EXCEPTION 'Aluno % estrangeiro precisa de um país do passaporte.', new.matricula;

RETURN NULL;

END IF;

IF new.validade\_passaporte IS NULL THEN

RAISE EXCEPTION 'Aluno % estrangeiro precisa de uma data de validade do passaporte.', new.matricula;

RETURN NULL;

END IF;

END IF;

RETURN NEW;

END;

$tg\_chk\_estrangeiro$ LANGUAGE plpgsql;

CREATE TRIGGER tg\_chk\_estrangeiro BEFORE INSERT OR UPDATE ON posgrad

FOR EACH ROW EXECUTE PROCEDURE chk\_estrangeiro();

1. Procedimentos

**Altera as “data\_fim” na tabela monitoria que estão como NULL para a data atual:**

CREATE OR REPLACE FUNCTION finaliza\_monitorias()

RETURNS void AS $$

BEGIN

UPDATE monitoria

SET data\_fim=current\_date

WHERE data\_fim IS NULL;

END;

$$ LANGUAGE plpgsql;

**Exibe quantos alunos de pós graduação estão em cada status da matrícula:**

CREATE OR REPLACE FUNCTION relatorio\_status()

RETURNS void as $$

DECLARE

status varchar(12);

num int;

tuplas cursor is

SELECT posgrad.status\_matricula, count(\*)

FROM posgrad

GROUP BY posgrad.status\_matricula;

BEGIN

OPEN tuplas;

RAISE INFO 'STATUS DA MATRICULA - QUANTIDADE';

loop

fetch tuplas into status, num;

if not found then exit;

end if;

RAISE INFO '% - %', status, num;

end loop;

CLOSE tuplas;

END;

$$ LANGUAGE plpgsql;

1. Funções

**Obtem o período a partir da data:**

CREATE OR REPLACE FUNCTION obtem\_periodo(data\_in date)

RETURNS numeric(5) AS $$

DECLARE

mes smallint;

ano smallint;

resultado numeric(4,1);

BEGIN

SELECT EXTRACT(YEAR FROM data\_in) into ano;

SELECT EXTRACT(MONTH FROM data\_in) into mes;

IF (mes <=6)THEN

resultado:=0.1;

ELSE

resultado:=0.2;

END IF;

RETURN resultado+ano;

END;

$$ LANGUAGE plpgsql;

**Obtem todos os alunos de pós graduação que se matricularam em um período específico:**

CREATE OR REPLACE FUNCTION obtem\_matriculados (periodo posgrad.periodoinicio%TYPE , tipopg posgrad.tipo%TYPE)

RETURNS bigint as $$

DECLARE

numero int;

BEGIN

SELECT COUNT (\*) into numero

FROM posgrad

WHERE posgrad.periodoinicio = periodo AND posgrad.tipo = tipopg;

RETURN numero;

END;

$$ LANGUAGE plpgsql;

1. Índices

Para o nosso banco de dados, prevemos que a maior parte das consultas seram feitas sobre matriculas e códigos de discilinas, os quais já são indexados implicitamente por serem chaves primárias. Na tabela posgrad, o CPF, link do currículo Lattes e login, também já possuem índices por serem UNIQUE. Analisamos que outras consultas que podem vir a ser relevantes são aquelas em cima da situação ou sttatus da matrícula de um pós-graduando e sobre as disciplinas de monitoria, assim criamos também os seguintes índices:

CREATE INDEX INX\_situacao ON posgrad (situacao)

CREATE INDEX INX\_status ON posgrad (status\_matricula)

CREATE INDEX INX\_discip\_monitoria ON monitoria (codigo)

1. Avaliação do Projeto

A qualidade do projeto lógico será avaliada através de dependências funcionais e formas normais.

**Dependências funcionais e formas normais:**

As dependências funcionais especificam propriedades de dados válidos no banco de dados.

X → Y

“X determina Y ou Y depende funcionalmente de X” sse

t1[X] = t2[X] ⇒ t1[Y] = t2[Y]

∀ tuplas t1, t2 em r instância de R.

A partir das dependências funcionais é possível determinar as chaves candidatas de cada tabela.

Sejam R (A1, A2,… An); F e X ⊆ {A1, A2,…An }

X é chave de R se X {A1, A2,… An } ∈ F+ e

não há Y ⊆ X tal que Y {A1, A2,… An } ∈ F+.

A normalização dos dados permite um armazenamento consistente e reduz a redundância de dados.

**Primeira Forma Normal (1FN):** Uma relação R está em 1FN se todos os atributos são atômicos/indivisíveis.

**Segunda Forma Normal (2FN):** Uma relação está em 2FN se estiver em 1FN e nenhum atributo não-primo depender funcionalmente de uma parte da chave.

**Terceira Forma Normal (3FN):** Uma relação está em 3FN se estiver em 2FN e todo atributo não primo depender apenas de atributos primos.

**Forma Normal de Boyce-Codd:** Uma relação R está em FNBC se toda DF não trivial de R X → A for tal que X é superchave de R.

**Aplicação no projeto lógico:**

**Aluno:**

- matricula → nome

Chave: matricula

Forma Normal de Boyce-Codd

**Grad:**

- matricula → curso

Chave: matricula

Forma Normal de Boyce-Codd

**Disciplina:**

- codigo → nome

Chave: codigo

Forma Normal de Boyce-Codd

**Laboratorio:**

- nome → área

- nome → responsável

- responsável → nome

Chaves: nome ; responsável

Chave escolhida: nome

Forma Normal de Boyce-Codd

**Bolsa:**

- {tipo,aluno} → data\_ini

- {tipo,aluno} → data\_fim

Chave: {tipo,aluno}

Forma Normal de Boyce-Codd

**Monitoria:**

- {aluno,codigo} → data\_ini

- {aluno,código} → data\_fim

Chave: {tipo,aluno}

Forma Normal de Boyce-Codd

**Estagio:**

- {aluno,laboratório} →data\_ini

- {aluno,laboratório} → data\_fim

Chave: {aluno,laboratório}

Forma Normal de Boyce-Codd

**Professor:**

Matricula → nome

Matricula → departamento

Matricula → nomelab

Matricula → data\_ini

Matricula → data\_fim

Chave: matricula

Forma Normal de Boyce-Codd

**Projetoorientado:**

- codigo → numcred

- {aluno,código} → tema

- {aluno,código} → numcred

- {aluno,código} → nota

- {aluno,código} → matriculaprof

- {aluno,código} → período

Chave: {aluno,código}

Terceira Forma Normal

**Posgrad:**

- CPF → matricula

- login → matricula

- CEP → logradouro

- CEP → cidade

- CEP → estado

- numagencia → nomeagencia

- matricula → todos os atributos

Segunda forma normal– a tabela não está na terceira forma normal porque o formulário de preenchimento de cadastro de aluno de pós-graduação foi seguido.