**Documentação SCA**

Conteúdo

[Modelo do Banco de Dados 3](#_Toc506811045)

[Tabela aluno 3](#_Toc506811046)

[Registro de aluno 4](#_Toc506811047)

[Tabela matricula 4](#_Toc506811048)

[Encerramento de matrícula 5](#_Toc506811049)

[Nova matrícula 5](#_Toc506811050)

[Tabela modalidade 5](#_Toc506811051)

[Tabela mensalidade 5](#_Toc506811052)

[Tabela matmod 6](#_Toc506811053)

[Tabela config 6](#_Toc506811054)

[Anotações 7](#_Toc506811055)

[Editar modalidades do aluno 7](#_Toc506811056)

[Filtrar alunos 8](#_Toc506811057)

[Módulos 9](#_Toc506811058)

[Métodos de alteração no banco de dados 10](#_Toc506811059)

[Inserção de aluno 11](#_Toc506811060)

[Encerramento de matricula 13](#_Toc506811061)

[Nova matrícula 14](#_Toc506811062)

[Remove aluno 15](#_Toc506811063)

[Remove matricula 16](#_Toc506811064)

[Inserção de modalidade 17](#_Toc506811065)

[Adição de modalidade ao aluno em MATMOD 18](#_Toc506811066)

[Encerra modalidade do aluno 18](#_Toc506811067)

[Finaliza modalidade 19](#_Toc506811068)

[Remove modalidade 19](#_Toc506811069)

[Inserção de pagamento 20](#_Toc506811070)

[Inserção de usuário 21](#_Toc506811071)

[Remoção de usuário 21](#_Toc506811072)

[Algoritmos 22](#_Toc506811073)

[Algoritmo do cadastro de aluno 22](#_Toc506811074)

# Modelo do Banco de Dados

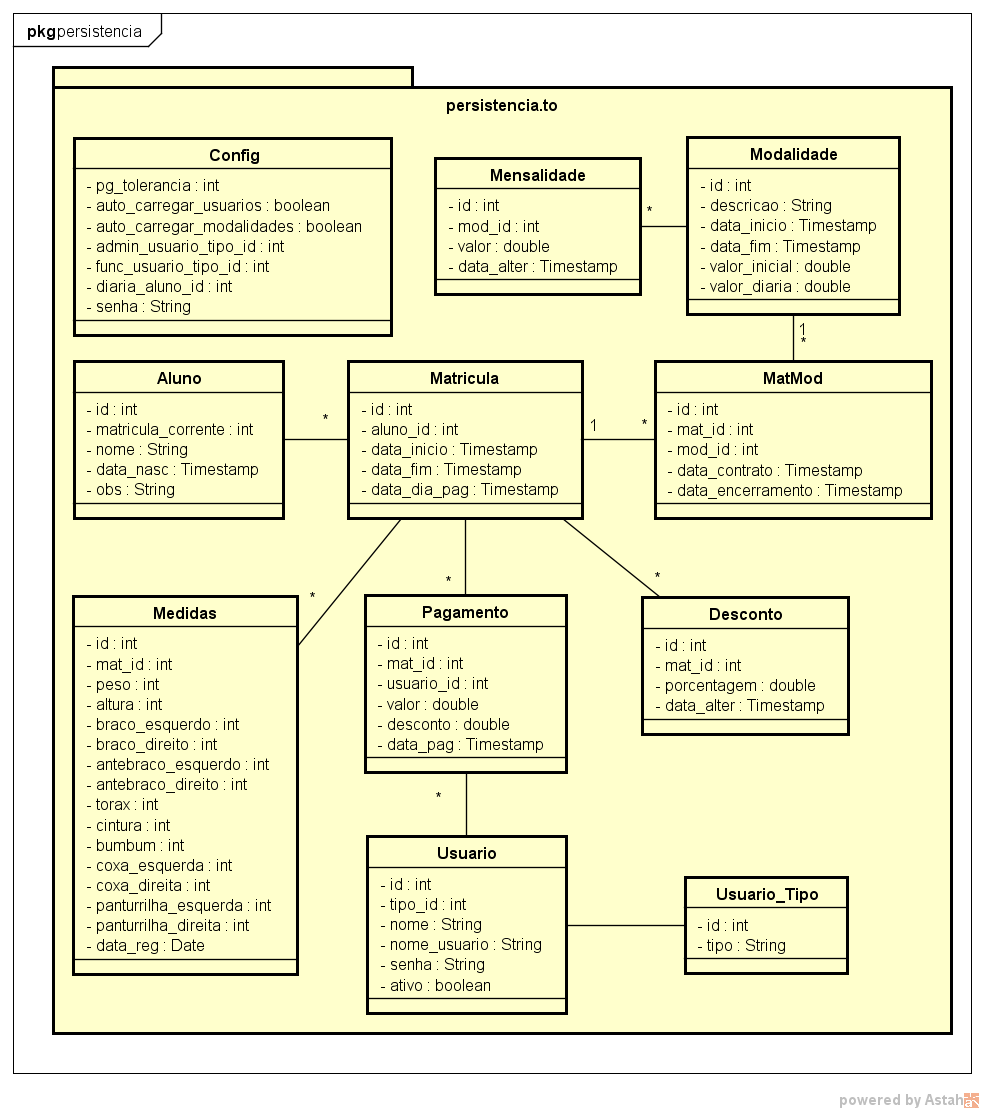


Figura 2 - Modelo do Banco de Dados

O modelo acima representa o modo que os dados estão persistidos em banco de dados. Abaixo uma breve descrição de cada tabela junto de seus campos e relação com outras tabelas:

## Tabela aluno

Na **tabela aluno** estão os campos: ***id, nome, data\_nasc, obs***. Onde, ***“id”*** é o identificador do aluno, **“nome”** é o nome do aluno, **“data\_nasc”** é a data de nascimento do aluno, **“obs”** é qualquer observação que se queira registrar e associar a um aluno.

O campo **“*matricula\_corrente”*** na tabela aluno representa a ultima matrícula registrada para o aluno, isto é, a matrícula corrente. É possível que o aluno tenha várias matrículas por causa das funções: **“Encerrar matrícula”** e **“Nova matrícula”**. A cada nova matrícula efetuada, uma nova matrícula é inserida no banco de dados com identificador do aluno (campo ***“aluno\_id”*** da tabela matricula) e o campo **“*matrícula\_corrente”*** passa a referenciar o identificador da nova matrícula criada.

### 

### Registro de aluno

Logo quando o aluno é registrado, é inserido em banco de dados o registro *na* ***tabela aluno***, além de inserido também um registro na ***tabela matricula*** e também é inseridos registros nas tabelas ***matmod*** a cada modalidade selecionada e, opcionalmente, na ***tabela medidas***. Então a **tabela aluno** é atualizada, tendo seu campo ***“matricula\_corrente”*** alterado para referenciar o identificador da matrícula registrada.

Obs: O aluno pode ter várias matrículas, porém, apenas uma ou nenhuma matrícula ativa.

## Tabela matricula

A tabela ***matricula*** no banco de dados mantém os seguintes campos de datas:

**data\_início:** Corresponde a data de registro da matrícula. Normalmente, a matrícula é inserida junto da inserção de um aluno ou depois de ser utilizada a função “**Nova matrícula**” para criar uma nova matrícula e atualizar o campo ***matricula\_corrente*** da tabela aluno para corresponder ao ID da nova matrícula adicionada.

**data\_fim:** Corresponde à data em que foi utilizada a função “**Encerrar matrícula**”. Com a execução da função “**Encerrar matrícula**”, o campo ***matricula\_corrente*** da tabela aluno é atualizado para **-1**, que é o código que representa: *nenhuma matrícula ativa*. Obs: Logo que uma matrícula é criada, o campo ***“data\_fim”*** torna-se **NULO**.

**data\_dia\_pag**: Corresponde a data do dia em que o aluno deve pagar. Se ***data\_inicio*** corresponder a uma data com dia maior que 28, ***data\_dia\_pag*** passa a ser igual ao primeiro dia do mês seguinte e, a mesma, fica sendo a data de pagamento pelo aluno.

Além dos campos de data a tabela matricula é composta também dos campos ***“id”*** e ***“aluno\_id”***, onde, ***“id”*** é o identificador da matricula e ***“aluno\_id”*** referencia o aluno associado à matrícula.

As tabelas ***medidas, pagamento, desconto, matmod*** referenciam a tabela matricula através do campo ***“aluno\_id”***.

### Encerramento de matrícula

Quando a função **“Encerrar matrícula”** é acionada, o sistema altera o campo ***“matricula\_corrente”*** da tabela ***aluno*** para não referenciar nenhuma matrícula, isto é, o valor **“-1”** é inserido no campo ***“matricula\_corrente”***. O campo ***“data\_fim”*** da tabela ***matricula*** é alterado para o valor da **data atual** no instante do encerramento da matrícula.

### Nova matrícula

Quando a função **“Nova matrícula”** é acionada, o sistema insere um novo registro de matrícula e atualiza o campo ***“matricula\_corrente”*** na tabela aluno para o mesmo referenciar o Identificador do novo registro de matrícula. Caso o aluno selecionado tenha uma matrícula, o sistema pergunta primeiro se deseja encerrar a matricula ativa para o aluno selecionado e, apenas se o usuário permitir, a matrícula é encerrada e uma nova é criada.

## Tabela modalidade

A tabela modalidade tem dois campos de data: ***“data\_inicio”*** e ***“data\_fim”***. O campo ***“data\_inicio”*** recebe a data de início da modalidade e o campo ***“data\_fim”*** recebe valor **NULO** até que a modalidade seja finalizada. O fim da modalidade acontece quando a função **“Finalizar modalidade”** é acionada.

O campo ***“*ID”** representa o identificador de cada registro de modalidade. O campo ***“descricao”*** deve receber como valor a descrição da modalidade. O campo ***“valor\_inicial”*** é o valor de início de contratação da modalidade e o campo ***“valor\_diaria”*** é o valor cobrado pelo uso da modalidade por um dia.

Ao inserir um **registro de modalidade**, o sistema insere também um **registro de mensalidade** com o campo valor igual ao campo ***“valor\_inicial”*** da tabela modalidade e o campo ***“data\_alter”*** igual ao campo ***“data\_inicio”*** da tabela modalidade. O campo “mod\_id” da tabela **mensalidade** é preenchido para referenciar o ***“ID”* da modalidade** que deve ser associada à mensalidade registrada.

## Tabela mensalidade

A **tabela mensalidade** tem os campos: ***id, mod\_id, valor, data\_alter***. Sempre que o valor da mensalidade é alterado, um novo registro é inserido na **tabela mensalidade**. Obs: para recuperação da mensalidade atual da modalidade, o sistema executa uma busca como a seguinte:

select \*

from mensalidade

where mod\_id=***”ID DA MODALIDADE”***

order by ( data\_alter ) desc

limit 1

O ***“id”*** é o identificador da mensalidade. O campo ***“mod\_id”*** é o numero que referencia a modalidade associada à mensalidade. O campo ***“valor”*** é o valor da mensalidade e o campo ***“data\_alter”*** é a data de alteração do valor da mensalidade.

## 

## Tabela matmod

A ***tabela matmod*** associa a modalidade à matrícula do aluno. Tem os campos: ***“id”*** que é o identificador único de cada registro, ***“mat\_id”*** que referencia a matrícula a ser associada a modalidade, ***“mod\_id”*** referencia a modalidade a ser associada a matrícula, ***“data\_contrato”*** é a data de contrato da modalidade e ***“data\_encerramento”,*** a data de encerramento da contratação da modalidade.

Logo quando um registro é inserido na tabela ***matmod***, o campo “***data\_encerramento“*** é registrado como **NULO**.

**Obs:** A ***tabela matmod*** pode ter registros com campos ***“mat\_id”*** e ***“mod\_id”*** iguais.

## Tabela config

A tabela **config** é composta também pelos campos a seguir:

**admin\_usuario\_tipo\_id:** Identificador do tipo de usuário que atribui ao aluno permissões de administrador.

**func\_usuario\_tipo\_id:** Identificador do tipo de usuário que atribui ao aluno permissões de funcionário.

**diaria.aluno.id:** Identificador que representa o aluno que recebe o pagamento de diária.

# Anotações

## Editar modalidades do aluno

Em Editar aluno é mostrada a seguinte tela:

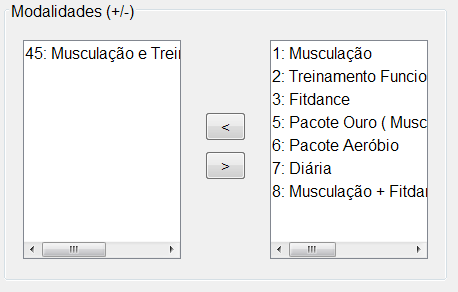


Figura 3 - Edição de modalidades contratadas

Onde, na **Figura 3**, as modalidades para adição estão na segunda lista (à direita) com seus respectivos IDs (Armazenados na tabela ***modalidade***) e, na primeira lista (à esquerda) estão às modalidades adicionadas a matrícula do aluno. Porém, o ID da modalidade adicionada mostrada na primeira lista é o ID da tabela no banco de dados ***matmod,*** tabela que representa as modalidades adicionadas à matrícula do aluno.

Para adicionar uma modalidade, o sistema extrai o código da(s) modalidade(s) selecionada(s) na lista à direita e insere um novo registro em ***matmod***. Obs: é possível ter contratos de modalidades registrados em ***matmod*** com mesmo *ID de matrícula* e *ID de modalidade*. Por isso a necessidade de utilizar o *ID de matmod* para a remoção lógica da modalidade adicionada ao aluno. No exemplo da **Figura 3** o *ID de matmod* é 45, não 4 que seria o ID da modalidade adicionada.

A remoção lógica é feita atualizando o campo ***data\_encerramento*** para corresponder a data atual no instante da execução remoção lógica.

Para encerrar a contratação de uma modalidade, o sistema extrai o código da modalidade selecionada na lista à esquerda e atualiza o registro da modalidade selecionada com alteração do campo ***data\_encerramento*** para data de fim do contrato (Data atual).

## Filtrar alunos

A função **“filtrar aluno”** é controlada por uma **variável** que define se os alunos que devem aparecer na tabela são os alunos da filtragem normal ou os alunos que estiverem completando ano.

A **variável** está na classe **academia.controlador.aluno.AlunoCtrlUtil** e é alterada nos seguintes métodos da classe **academia.controlador.aluno.FiltroAlunoControlador**:

* filtrarBTAcionado
* mostrarAlunosInativosCBAlterado
* aniverBTAcionado

# Módulos

Pasta “**bd**”: Módulo de acesso e manipulação de dados no SGBD.

Pasta “**gui**”: Módulo onde fica as classes de interface gráfica com o usuário.

Pasta “**controlador**”: Módulo onde fica a implementação de ações iniciadas através da interface gráfica. O padrão de projetos “observer” foi implementado para separar a implementação de interface gráfica das implementações de ações de componentes de interface gráfica.

**EX**: **AlunoControlador** implementa os métodos da interface **AlunoGUIListener**. E é registrado através do método **setAlunoGUIListener** da classe **AlunoGUI**.

Pasta “**util**”: Onde ficam as classes utilitárias de manipulação de datas, de números de ponto flutuante e de valores de moeda Real.

Pasta “**loginoper**”: Módulo onde ficam as implementações de operações que solicitam login mesmo após o usuário autenticado.

Pasta “**compmed**”: Módulo onde fica a classe que é utilizada na comparação de medidas. O sistema permite a comparação de medidas registradas em datas diferentes.

Pasta “**config**”: Módulo onde fica a leitura e carregamento do arquivo de configuração.

Pasta “**relatorio**”: Módulo onde ficam as implementações dos relatórios do dia, relatório de alunos em dia e comparação de medidas. A implementação dos relatórios é acessada através de uma classe de fachada chamada de **GerenciadorRelatorio**.

# Métodos de alteração no banco de dados

## Inserção de aluno

A inserção de aluno no banco de dados, afeta as tabelas: **aluno** e **matricula**. Dependendo da configuração pode afetar também as tabelas **medidas, matmod e desconto.**

**Instruções SQL no método insere da classe AlunoDAO:**

**Variáveis:**

**$aluno\_id** = **nextval**( ‘**aluno\_seq**’ );

**$mat\_id** = **nextval**( ‘**matricula\_seq**’ );

**$nome** = ‘nome do aluno’;

**$data\_nasc** = ‘data de nascimento do aluno’;

**$obs** = ‘Observação sobre o aluno’;

**$data\_mat** = ‘data de matrícula do aluno’;

**$data\_dia\_pag** = ‘dia de pagamento entre 1 e 28 dada a possibilidade de haver anos em que no mês de fevereiro tem apenas 28 dias. Caso a **$data\_mat** tenha como dia valor maior que 28, **$data\_dia\_pag** assume o valor 1’;

**Instruções SQL:**

insert into **ALUNO** ( **id, matricula\_corrente, nome, data\_nasc, obs** ) values (

**$aluno\_id,**

**$mat\_id,**

**$nome,**

**$data\_nasc,**

**$obs**

);

insert into **MATRICULA** ( **id, aluno\_id, data\_inicio, data\_dia\_pag** ) values (

**$mat\_id,**

**$aluno\_id**

**$data\_mat,**

**$data\_dia\_pag**

);

insert into **DESCONTO** ( **mat\_id, porcentagem, data\_alter** ) values (

**currval( ‘matricula\_seq’ ),**

**$porcentagem,**

**$data\_mat**

);

Se houver modalidades para adicionar {

Para cada modalidade fazer {

**$mod\_id** = ‘id da modalidade a ser adicionada’;

Insert into **MATMOD** ( **mat\_id, mod\_id, data\_contrato** ) values (

**$mat\_id**,

**$mod\_id,**

**$data\_mat**

);

}

}

Se houver medidas para adicionar {

insert into **MEDIDAS** ( **mat\_id, ...** ) values ( **$mat\_id, …** );

}

## Encerramento de matricula

O enceramento de matrícula afeta as tabelas: **aluno** e **matricula**.

**Instruções SQL do método encerraMat da classe AlunoDAO:**

**Variáveis**:

**$aluno\_id** = ‘Identificador do aluno’;

**$mat\_corr\_id** = ‘Matrícula corrente do aluno com id igual a **$aluno\_id**’;

**$data\_fim** = ‘Data corrente atual’;

**Instruções SQL:**

update **MATRICULA** set

**data\_fim** = **$data\_fim**

where **id** = **$mat\_corr\_id**;

update **ALUNO** set

**matricula\_corrente** = **NULL** **🡪-1**

where **id** = **$aluno\_id**;

## Nova matrícula

A criação de nova matrícula afeta as tabelas: **aluno** e **matrícula**.

**Instruções SQL do método novaMat da classe AlunoDAO:**

**Variáveis:**

**$aluno\_id** = ‘ID do aluno’;

**$mat\_corr\_id** = ‘ID da matrícula corrente’;

**$data\_mat** = ‘Data de registro da nova matrícula (Data corrente atual);

**$data\_dia\_pag** = ‘dia de pagamento entre 1 e 28 dada a possibilidade de haver anos em que no mês de fevereiro tem apenas 28 dias. Caso a **$data\_mat** tenha como dia valor maior que 28, **$data\_dia\_pag** assume o valor 1’;

**Instruções SQL:**

Se matrícula não encerrada ( $mat\_corr\_id <> -1 ) {

update **MATRICULA** set

**data\_fim** = **current\_timestamp**

where **id** = **$mat\_corr\_id**;

}

insert into **MATRICULA** ( **aluno\_id, data\_inicio, data\_dia\_pag** ) values (

**$aluno\_id,**

**$data\_mat,**

**$data\_dia\_pag**

);

update **ALUNO** set

**matricula\_corrente** = **currval( ‘matricula\_seq’ )**

where **id** = **$aluno\_id;**

## Remove aluno

A remoção de aluno afeta as tabelas: **aluno, matricula, desconto, pagamento, medidas, matmod**.

Tabela **MATRICULA** (

...,

Foreign key ( **aluno\_id** ) references aluno( id ) **on delete cascade**

);

Tabela **DESCONTO** (

...,

Foreign key ( **mat\_id** ) references matricula( id ) **on delete cascade**

);

Tabela **PAGAMENTO** (

...,

Foreign key( **mat\_id** ) references matricula( id ) **on delete cascade**

);

Tabela **MEDIDAS** (

...,

Foreign key( **mat\_id** ) references matricula( id ) **on delete cascade**

);

Tabela **MATMOD** (

...,

Foreign key( **mat\_id** ) references matricula( id ) on delete cascade

);

**Instruções SQL do método remove da classe AlunoDAO**

**Variáveis:**

**$aluno\_id** = ‘ID do aluno’;

**Instrução SQL:**

delete from **ALUNO** where **id** = **$aluno\_id;**

## Remove matricula

A remoção de matrícula afeta as tabelas: **matricula, pagamento, desconto, matmod e medidas**.

Tabela **DESCONTO** (

...,

Foreign key ( **mat\_id** ) references matricula( id ) **on delete cascade**

);

Tabela **PAGAMENTO** (

...,

Foreign key( **mat\_id** ) references matricula( id ) **on delete cascade**

);

Tabela **MEDIDAS** (

...,

Foreign key( **mat\_id** ) references matricula( id ) **on delete cascade**

);

Tabela **MATMOD** (

...,

Foreign key( **mat\_id** ) references matricula( id ) on delete cascade

);

**Instruções SQL do método remove da classe MatriculaDAO**

**Variáveis:**

**$mat\_id** = ‘ID da matrícula a ser removida’;

**Instrução SQL:**

delete from **MATRICULA** where **id** = **$mat\_id**;

## Inserção de modalidade

O cadastro de modalidade pode afetar as seguintes tabelas: **modalidade** e **mensalidade**.

**Instruções SQL do método insere da classe ModalidadeDAO:**

**Variáveis:**

**$descricao** = ‘Descrição da modalidade’;

**$valor\_inicial** = ‘Valor inicial da mensalidade’;

**$valor\_diaria** = ‘Valor por diária’;

**$data\_inicio** = ‘Data em que a modalidade passou a ser oferecida para os alunos’;

**Instruções SQL:**

Insert into **MODALIDADE** ( **descricao, valor\_inicial, valor\_diaria, data\_inicio** ) values (

**$descricao,**

**$valor\_inicial,**

**$valor\_diaria,**

**$data\_inicio**

);

insert into **MENSALIDADE** ( **mod\_id, valor, data\_alter** ) values (

**currval( ‘modalidade\_seq’ ),**

**$valor\_inicial,**

**$data\_inicio**

);

## Adição de modalidade ao aluno em MATMOD

A adição de modalidade afeta apenas a tabela **matmod**.

**Instruções SQL do método adicionaModalidadeAoAluno da classe ModalidadeDAO**

**Variáveis:**

**$mat\_id** = ‘ID da matrícula’;

**$mod\_id** = ‘ID da modalidade’;

**Instrução SQL:**

Insert into **MATMOD** ( **mat\_id, mod\_id** ) values ( **$mat\_id, $mod\_id** );

## Encerra modalidade do aluno

A adição de modalidade afeta apenas a tabela **matmod**.

**Instruções SQL do método adicionaModalidadeAoAluno da classe ModalidadeDAO**

**Variáveis:**

**$matmod\_id** = ‘ID da associação entre matrícula e modalidade (tabela MATMOD )

**Instrução SQL:**

update **MATMOD** set

**data\_encerramento** = **current\_timestamp**

where **id** = **$matmod\_id**;

## Finaliza modalidade

Método utilizado para marcar a modalidade como finalizada, **setando** o valor de “**data\_fim**” para data de finalização (data atual). Enquanto a modalidade está ativa, “**data\_fim**” é igual à NULL.

A finalização de modalidade afeta apenas a tabela **modalidade**.

**Instruções SQL do método finalizar da classe ModalidadeDAO**

**Variáveis:**

**$mod\_id** = ‘ID da modalidade a ser finalizada’;

**Instrução SQL:**

Update **MODALIDADE** set

**data\_fim** = **current\_timestamp**

where **id** = **$mod\_id**;

## Remove modalidade

A remoção de modalidades afeta as tabelas: **modalidade** e **mensalidade, matmod.**

Tabela **MENSALIDADE** (

...,

Foreign key( **mod\_id** ) references modalidade( id ) **on delete cascade**

);

Tabela **MATMOD** (

...,

Foreign key( **mod\_id** ) refererences modalidade( id ) **on delete cascade**

);

**Instrução SQL do método remove da classe ModalidadeDAO**

**Variáveis:**

**$mod\_id** = ‘ID da modalidade a ser removida’;

**Instrução SQL:**

Delete from **MODALIDADE** where **id** = **$mod\_id**;

## Inserção de pagamento

**Observação sobre o cadastro de aluno:** Apenas se a opção inserir pagamento estiver selecionada na tela de cadastro de aluno, a função de **inserir** da classe **PagamentoDAO** é executada independente do método inserir da classe **AlunoDAO**. Na inserção de pagamento junto do cadastro de aluno, o valor do **desconto associado ao pagamento** (Desconto dado durante o cadastro de um pagamento) é igual a zero.

**Instrução SQL do método insere da classe PagamentoDAO sem desconto associado ao pagamento:**

**Variáveis:**

**$mat\_id** = ‘ID da matrícula corrente do aluno ao qual o pagamento é associado’;

**$usuario\_logado\_id** = ‘ID do usuário logado no sistema no instante do cadastro do pagamento’;

**$valor** = ‘Valor calculado com **desconto associado ao aluno** (Desconto informado no formulário de cadastro ou alteração de aluno) e com **desconto associado ao pagamento**.

**$desconto** = ‘**desconto associado ao pagamento**. **OBSERVAÇÃO**: Na inserção de pagamento com o **CADASTRO DE ALUNO**, o valor do desconto é **IGUAL A ZERO’**;

**$data\_pag** = ‘Data de matrícula do aluno a ser cadastrado’;

**Código SQL:**

Insert into **PAGAMENTO** ( **mat\_id, usuario\_id, valor, desconto, data\_pag** ) values (

$mat\_id, $usuario\_logado\_id, $valor, $desconto, $data\_pag

);

## Inserção de usuário

A inserção de usuários afeta apenas a tabela **usuário.**

**Instruções SQL do método insere da classe UsuarioDAO**

**Variáveis:**

**$nome** = ‘Nome do usuário a ser cadastrado’;

**$nome\_usuario** = ‘Nome de usuário para ser utilizado na autenticação’;

**$senha** = ‘Senha do usuário para autenticação’;

**$tipo\_id** = ‘ID do tipo de usuário’;

**Instruções SQL:**

Insert into **USUARIO** ( **nome, nome\_usuario, senha, tipo\_id** ) values (

**$nome,**

**$nome\_usuario,**

**md5( $senha ),**

**$tipo\_id**

);

## Remoção de usuário

A remoção de usuário afeta apenas a tabela **usuário**.

Em caso de o usuário ter pagamentos registrados por ele à remoção não é permitida, pois, caso contrário, uma exceção “**SQLException**” seria gerada.

**Instruções SQL do método remove da classe UsuarioDAO**

**Variáveis:**

**$usuario\_id** = ‘ID do usuário’;

**Instrução SQL:**

delete from **USUARIO** where id = **$usuario\_id**;

# Algoritmos

## Algoritmo do cadastro de aluno

**$data\_pag** = **calcula\_data\_pag**( **$data\_mat** );

**$medidas** = **carrega\_objeto\_medidas**();

/\*\* se função carregar medidas estiver habilitada e a opção pular medidas não tiver sido selecionada \*\*/

**$lista\_IDs** = **carrega\_lista\_de\_IDs\_das\_modalidades**();

Se houver alguma modalidade adicionada {

**Carrega Lista de IDs das modalidades** do seguinte modo:

Para cada modalidade {

**$id** = **extrairModalidadeID**( **$mod\_id\_str** );

/\*\* Os IDs selecionados estão como string no formato

“**ID: Modalidade**”. Ex: “**9: Musculação**”, nesse exemplo o valor extraído equivale a **9**; \*\*/

}

}

}

**$aluno** = **carrega\_objeto\_aluno**( **$data\_mat, $data\_pag, $medidas, $lista\_IDs**, ... );

**insere**( **$aluno** );

se **inserir\_pagamento** = **SIM** então {

**$modalidades** = **busca\_modalidades\_do\_aluno**( **$aluno\_id** );

**$parcela** = **gera\_parcela**( **$modalidades, $data\_pag, $desconto, $desconto\_pag** );

/\*\* **$desconto\_pag** igual a 0 \*\*/

**$valor** = **get\_subtotal\_sem\_desconto\_pag( $parcela** );

Ou

**$valor** = **get\_valor\_total**( **$parcela** );

/\*\* isso porque o valor $desconto\_pag é igual a 0 \*\*/

**$pag** = **carrega\_pagamento**( **$mat\_id, $usuario\_logado\_id, $data\_pag, $valor** );

**insere\_pagamento**( **$pag** ); /\*\*Definido anteriormente \*\*/

}