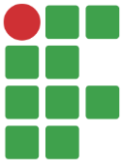

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DA PARAÍBA
ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

JOZIMAR SOARES DA COSTA
RÔMULO SOARES BEZERRA

SISTEMA DE GERENCIAMENTO DE ACADEMIA FITNESS

Cajazeiras
2017

JOZIMAR SOARES DA COSTA
RÔMULO SOARES BEZERRA

SISTEMA DE GERENCIAMENTO DE ACADEMIA FITNESS

Trabalho apresentado ao Curso Superior Tecnológico em Análise e Desenvolvimento de Sistemas do IFPB – Instituto Federal de Educação, Ciência e Tecnologia da Paraíba, para a disciplina Banco de Dados.

Prof. Dr. Fabio Gomes de Andrade

Cajazeiras
2017

Sumário

1. Introdução

Com extensão das academias de musculação e o avanço tecnológico, gerir um negócio tão abrangente sem o auxílio da tecnologia se tornou uma atividade inviável. Com base nisso, no intuito de prevenir más ocorrências evitando possíveis dores de cabeça com o descaminho dos dados, o dono da Top Fitness Academia requisitou um sistema que o ajude a otimizar o gerenciamento de seu estabelecimento - informações de clientes, negociação, gastos, etc. Todos os seus dados e de seus alunos são armazenados em fichas de papelão que estão sujeitas a deterioração por agentes naturais (poeira e umidade) que podem ocasionar a perda dessas informações e gerar um retrabalho excessivo e desagradável.

MODELAGEM CONCEITUAL

SISTEMA DE GERENCIAMENTO DE ACADEMIA FITNESS

2.1 Levantamento de Requisitos

O sistema precisa cadastrar alunos. Do aluno precisam ser armazenados: nome, sexo, telefone, peso, idade, altura, IMC, BF, estado de saúde e objetivo. Também precisam ser armazenadas informações extras do aluno com relação a mensalidade. Em mensalidade são registradas o valor, data de pagamento e data de recibo em que o aluno fez o pagamento. O aluno tem também matricula que possui código, data de abertura e data de trancamento.

Ao se cadastrar, o aluno, é feita uma mensuração de medidas corporais que são vistas a cada mês, guardando-se a data, desde a data de entrada, para o acompanhamento de seu desenvolvimento. As medidas corporais do aluno são ponderadas em centímetros e compostas por medidas de braços: esquerdo e direito, antebraços: esquedo e direito, deltoides, peitoral, abdome, coxas: esquerda e direita e panturrilhas: esquerda e direita.

A academia possui funcionários que podem ser classificados em dois tipos: gerente que administra todo o empreendimento e/ou professor. Para professor e gerente são mantidos: nome, sexo, CPF e data de pagamento do salário. Em especificação, professor tem especialidade (s) e salário.

O gerente gerencia despesas, com descrição, valor e data, que são geradas a partir de necessidades integrantes. O gerente pode realizar negociações com o aluno, podendo o aluno adquirir produtos como por exemplo suplementação e gourmet para consumo ou gerente para consumo e revenda, sendo necessário guardar a data da realização. É importante armazenar informações da negociação também como: descrição, valor, CNPJ ou CPF. As negociações podem ser a vista ou a prazo, do tipo compra ou venda e possuem pagamento guardando-se o desconto (caso haja), quantidade de parcelas, valor de cada parcela e um atributo para verificar se foi quitada. Para produto, devem ser guardados o nome, marca, quantidade e preço.

Para cada aluno é montado um treino e exercícios pelo professor. Cada treino tem descrição, uso de carga, o tipo - que é a combinação dos tipos distintos trabalhados nos exercícios, tempo de intervalo de descanso em dias quando completo o ciclo do tipo, tempo de intervalo entre as séries, tempo de intervalo entre as

repetições, data de início e duração até a mudança de treino.

Exemplo: - O aluno Abcelino está com o treino descrito como: útil para hipertrofia muscular, com uso de carga pesada, tipo ABC, com intervalo de descanso: um dia após a realização dos exercícios da divisão “C” para novamente executar o treino “A”, 3 minutos de descanso para execução de uma nova série, com 30 segundos entre as repetições. Ele está trabalhando uma ou duas vezes por semana o mesmo músculo. Iniciou o treino dia 24/02/2017, e tem como duração 2 meses até a troca ou adaptação do novo treino.

O treino tem exercícios que são particulares a grupos musculares. Os exercícios têm músculo trabalhado, nome, número de séries, quantidade de repetições, se é Drop Set e o tipo, se: A, B, C, D, E ou F. Cada tipo é uma divisão de exercícios de grupos musculares e treinado por dia. Os treinos e os exercícios são montados e supervisionados pelo professor.

Logo, concluindo com o exemplo de Abcelino; ele numa segunda-feira vai executar o primeiro exercício para bíceps braquial de nome “rosca direta” com 3 vezes de 12 repetições, não Drop Set e referente a sigla “A” do tipo de treino “ABC”. Como segundo exercício, também para bíceps braquial ele fará a “rosca martelo”, com 4 vezes de 10 repetições, é Drop Set e também de “A” já que é praticado no mesmo dia. Terminado esses exercícios, Abcelino agora vai malhar o exercício para peitoral, com nome supino reto, 3 de 15, não Drop Set e de “A”, e em seguida com mais um de peitoral “supino declinado com halteres”, 4 de 8, não Drop Set, e de “A”.

Para melhor entendimento, o aluno Abcelino numa terça-feira vai malhar um exercício para dorsal “remada baixa”, com 4 vezes de 12 repetições, não Drop Set e referente a sigla do dia “B”. Como segundo exercício: “remada curvada”, 3 de 10, Drop Set e da sigla do tipo “B”. E assim segue o mesmo para a quarta-feira; cada dia terá vários exercícios trabalhando músculos com características e execuções diferentes.

2.2 Modelo Conceitual

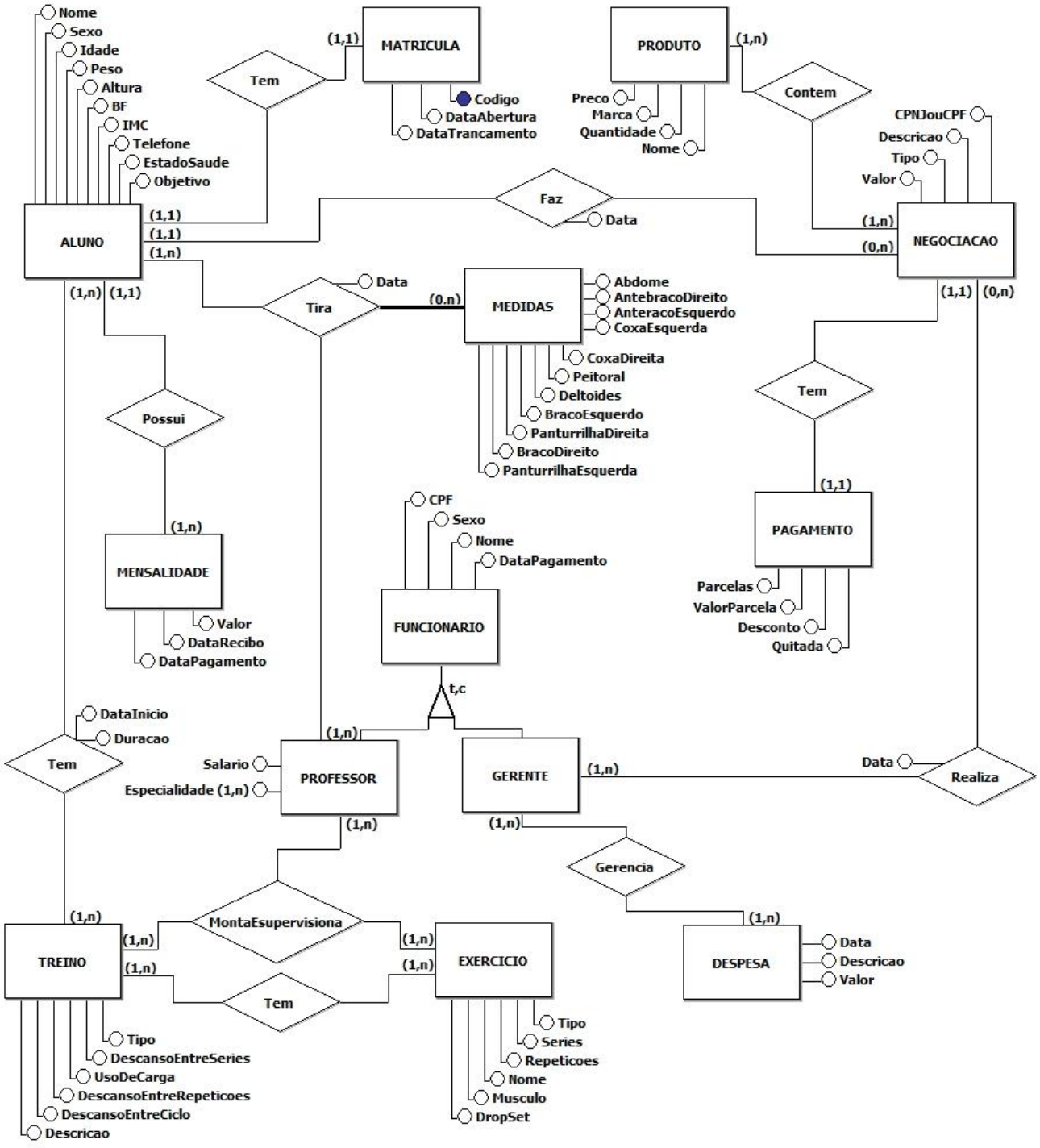

Figura 1: Modelo Conceitual

2.3 Dicionário Conceitual de Dados

Entidade ALUNO

- ALUNO: criada para guarda informações sobre aluno.

Atributo (s):

- Nome: criado para armazenar o nome de cada aluno.
- Sexo: criado para guardar o sexo de um determinado aluno.
- Idade: criado para guardar a idade de um determinado aluno.
- Peso: criado para manter o dado do peso de um aluno.
- Altura: criado para manter a altura de um aluno.
- BF: derivado, criado para armazenar o índice da gordura corporal (*body fat)* do aluno.
- IMC: derivado, criado para reservar o índice da massa corpórea de um aluno.
- Telefone: criado para guardar o número de telefone do aluno.
- EstadoSaude: derivado, criado para guardar a descrição do estado de saúde de um aluno de acordo com o IMC: abaixo do peso, peso normal, acima do peso...
- Objetivo: criado para guardar a informação do objetivo do aluno: ganho de massa muscular, definição muscular, emagrecimento...

Relacionamento (s):

- Tem: relaciona Aluno com Matricula. Um aluno tem somente uma matrícula assim como uma determinada matricula pode pertencer a apenas um aluno.
- Faz: relaciona Aluno com a entidade Negociacao. Cada aluno pode ou não fazer uma ou várias negociações e uma dada negociação é feita por somente um aluno, guardando-se a data de realização.
- Tira: relaciona Aluno à Professor e Medidas. Do aluno pode-se ou não tirar uma ou mais medidas por um ou mais professores e um professor pode ou não tirar uma ou mais medidas de um ou mais alunos, guardando-se a data da mensuração.
- Possui: relaciona Aluno com Mensalidade. Um aluno tem uma ou várias mensalidades (a cada mês uma nova) e cada mensalidade pertence a

apenas um aluno.

- Tem: relaciona Aluno à Treino. Um aluno tem um ou vários treinos e um treino pode pertencer a um ou mais alunos, guardando a data de início e duração do treino.

Entidade MATRICULA

- MATRICULA: criada para guardar informações de matricula de aluno.

Atributo (s):

- DataAbertura: criado para guardar a data da abertura da matrícula.
- Codigo: criado para manter o número da matricula.
- DataTrancamento: criado para guardar a data de trancamento da matrícula.

Relacionamento (s):

- Tem: relaciona Matricula à entidade Aluno. Uma matricula pode pertencer a unicamente um aluno e um aluno tem unicamente uma matricula.

Entidade MENSALIDADE

- MENSALIDADE: criada para guardar informações de mensalidades de aluno.

Atributo (s):

- Valor: criado para manter o valor da mensalidade.
- DataRecibo: criado para armazenar a data de recebimento do pagamento da mensalidade.
- DataPagamento: criado para guardar a próxima data de pagamento da mensalidade.

Relacionamento (s):

- Possui: relaciona Mensalidade à entidade Aluno. Uma mensalidade pertence a um único aluno e um determinado aluno tem uma ou muitas mensalidades (a cada mês uma nova).

Entidade TREINO

- TREINO: criado para guardar informações de treinos.

Atributo (s):

- Tipo: criado para armazenar informação do tipo de treino do aluno: AB, ABC ou ABCD...
- DescansoEntreSeries: criado para guardar o tempo de descanso entre cada série.
- UsoDeCarga: criado para armazenar o uso de carga padrão do treino: leve, moderada ou pesada.
- DescansoEntreRepeticoes: criado para guardar o tempo de descanso entre repetições do treino.
- DescansoEntreCiclo: criado para armazenar os dias de descanso ao término de um ciclo do tipo do treino.
- Descricao: criado para armazenar a descrição do treino: ganho de massa magra, definição...

Relacionamento (s):

- MontaEsupervisiona: relaciona Treino a entidade Professor e Exercício. O treino é montado e supervisionado por um ou mais professores que possui um ou mais exercícios e os exercícios são montados e supervisionados por um ou mais professores e estão presentes em um ou mais treinos.
- Tem: relaciona Treino a entidade Exercício. Um treino tem um ou muitos exercícios, assim como um exercício está contido em um ou mais treinos.
- Tem: relaciona Treino à Aluno. Um treino é tido por um ou mais alunos, bem como alunos têm um ou mais treinos guardando a data de início de prática do treino e a duração.

Entidade EXERCICIO

- EXERCICIO: criado para guardar informações de exercícios.

Atributo (s):

- Tipo: criado para guardar a sigla do dia do exercício executado: A, B, ou C...
- Series: criado para armazenar o número de séries de um determinado exercício.
- Repeticoes: criado para armazenar o número de repetições da série de um dado exercício.

- Nome: criado para guardar o nome do exercício.
- Musculo: criado para armazenar o nome do musculo trabalhado pelo exercício.
- DropSet: criado para armazenar uma informação da forma de execução do exercício: se Drop Set ou não.

Relacionamento (s):

- MontaEsupervisiona: relaciona Exercicio à entidade Treino e Professor. Um exercício é montado e supervisionado por um ou mais professores e são pertencentes a um ou muitos treinos e os treinos têm um ou muitos exercícios que são montados por um ou mais professores.
- Tem: relaciona Exercicio à Treino. Os exercícios estão presentes em um ou muitos treinos, assim como os treinos possuem um ou mais exercícios.

Entidade FUNCIONARIO

- FUNCIONARIO: generalização das entidades Professor e Gerente, criada para guardar informações gerais do funcionário.

Atributo (s):

- CPF: criado para guardar o número do CPF do funcionário.
- Sexo: criado para guardar a informação do sexo do funcionário.
- Nome: criado para armazenar o nome do funcionário.
- DataPagamento: criado para guardar a data do pagamento do salário do funcionário.

Relacionamento (s):

- *Sem relacionamento.*

Entidade PROFESSOR

- PROFESSOR: especialização da entidade Funcionario criada para especificar e guardar informações do professor (tipo de funcionário).

Atributo (s):

- Salario: criado para guardar a informação do valor do salário do professor.
- Especialidade: criado para guardar as especialidades do professor.

Relacionamento (s):

- Tira: relaciona Professor à entidade Aluno e à Medidas. Um professor pode ou não tirar uma ou mais medidas de um ou mais alunos, todavia as medidas de um ou mais alunos são tiradas por um ou mais professores, guarda-se a data da mensuração.
- MontaEsupervisiona: relaciona Professor com a entidade Treino e Exercício. Um professor monta e supervisiona um ou mais treinos com um ou mais exercícios e os exercícios são montados e supervisionados por um ou mais professores que estão presentes em um ou mais treinos.

Entidade GERENTE

- GERENTE: especialização da entidade Funcionario criada para especificar e guardar informações do gerente (tipo de funcionário).

Atributo (s):

- *Sem atributo.*

Relacionamento (s):

- Gerencia: relaciona Gerente à entidade Despesa. Um gerente gerencia uma ou várias despesas e as despesas são geridas por um ou mais gerentes.
- Realiza: relaciona Gerente com Negociacao. Um gerente pode ou não realizar uma ou várias negociações e as negociações são feitas por um ou mais gerentes, guardando-se a data da realização.

Entidade DESPESA

- DESPESA: criada para guardar informações de despesa.

Atributo (s):

- Descricao: criado para armazenar a descrição da despesa.
- Valor: criado para armazenar o valor de determinada despesa.
- Data: criado para reservar a data em que a despesa foi adquirida.

Relacionamento (s):

- Gerencia: relaciona Despesa a entidade Gerente. Uma despesa é gerida por um ou vários gerentes e um gerente gerencia uma ou várias despesas.

Entidade NEGOCIACAO

- NEGOCIACAO: criada para guardar informações gerais da negociação.

Atributo (s):

- CNPJouCPF: criado para armazenar o CNPJ do fornecedor de uma negociação feita pelo gerente, ou CPF do aluno que fez a negociação.
- Descricao: criado para manter dados da descrição da negociação.
- Valor: criado para guardar o valor de uma determinada negociação.
- Tipo: criado para guardar a espécie da negociação.

Relacionamento (s):

- Faz: relaciona Negociacao com a entidade Aluno. Uma negociação é feita por um e somente um aluno e uma aluno faz ou não uma ou muitas negociações, guardando a data da realização.
- Contem: relaciona Negociacao à Produto. Uma negociação possui um ou mais produtos e um produto pode estar presente em uma ou mais negociações.
- Realiza: relaciona Negociacao à entidade Gerente. Uma negociação pode ser realizada por um ou vários gerentes e um gerente pode fazer zero ou muitas negociações, guardando-se a data da negociação.

Entidade PAGAMENTO

- PAGAMENTO: criada para guardar informações de pagamento de uma negociação.

Atributo (s):

- ValorParcela: criado para guardar o valor das parcelas de cada negociação.
- Parcelas: criado para guardar o número de parcelas em que uma negociação foi dividida.
- Quitada: criado para armazenar a informação de quitação do valor de todas as parcelas de uma negociação, se totalmente paga ou não.
- Desconto: criado para guardar o valor do desconto de uma determinada negociação.

Relacionamento (s):

- *Sem relacionamento.*

Entidade PRODUTO

- PRODUTO: criada para guardar informações de produto.

Atributo (s):

- Preco: criado para manter o preço de um determinado produto.
- Quantidade: criado para guardar a quantidade de um mesmo produto.
- Nome: criado para guardar o nome de um dado produto.
- Marca: criado para guardar a marca de um certo produto.

Relacionamento (s):

- Contém: relaciona Produto à entidade Negociacao. Um ou mais produtos pode estar presente em uma ou mais negociações e uma negociação pode conter um ou mais produtos.

Entidade MEDIDAS

- MEDIDAS: criada para guardar informações sobre medidas de aluno.

Atributo (s):

- Peitoral: criado para guardar a informação da medida em centímetros do peitoral.
- Abdome: criado para guardar informação da medida em centímetros do abdome.
- Deltoides: criado para guardar a informação das medidas em centímetros dos deltoides.
- CoxaEsquerda: criado para armazenar a informação da medida em centímetros da coxa esquerda.
- CoxaDireita: criado para armazenar a informação da medida em centímetros da coxa direita.
- AntebracoEsquerdo: criado para guardar o dado da medida em centímetros do antebraco esquerdo.
- AntebracoDireito: criado para guardar o dado da medida em centímetros do

antebraco direito.

- BracoEsquerdo: criado para armazenar o dado da medida em centímetros do braço esquerdo.
- BracoDireito: criado para armazenar o dado da medida em centímetros do braço direito.
- PanturrilhaEsquerda: criado para guardar o dado da medida em centímetros da panturrilha esquerda.
- PanturrilhaDireita: criado para guardar o dado da medida em centímetros da panturrilha direita.

Relacionamento (s):

- Tira: relaciona Medida à entidade Aluno e Professor. As medidas de um ou mais alunos são feitas por um ou mais professores e um professor pode ou não tirar uma ou mais medidas de um ou mais alunos, guarda-se a data da realização.

MODELAGEM LÓGICA

SISTEMA DE GERENCIAMENTO DE ACADEMIA FITNESS

3.1 Mapeamento Entidade Relacionamento

ALUNO (Codigo, Nome, Sexo, Idade, Peso, Altura, BF, IMC, Telefone, EstadoSaude, Objetivo);
MATRICULA (Codigo, CodigoAluno, DataAbertura, DataTrancamento);
MENSALIDADE (Codigo, CodigoAluno, Valor, DataRecibo, DataPagamento);
TREINO (Codigo, Tipo, DescansoEntreSeries, UsoDeCarga, DescansoEntreRepeticoes, DescansoEntreCiclo, Descricao);
ALUNO_Tem_TREINO (CodigoAluno, CodigoTreino, Duracao, DataInicio);
EXERCICIO (Codigo, Tipo, Series, Repeticoes, Nome, Musculo, DropSet);
TREINO_Tem_EXERCICIO (CodigoTreino, CodigoExercicio);
PROFESSOR (Codigo, CPF, Sexo, Nome, DataPagamento, Salario);
EspecialidadePROFESSOR (CodigoProfessor, Especialidade);
PROFESSOR_EXERCICIO (CodigoProfessor, CodigoTreino, CodigoExercicio);
GERENTE (Codigo, CPF, Sexo, Nome, DataPagamento);
DESPESA (Codigo, Descricao, Valor, Data);
GERENTE_Gerencia_DESPESA (CodigoGerente, CodigoDespesa);
NEGOCIACAO (Codigo, CodigoAluno, CNPJouCPF, Descricao, Valor, Tipo, Data);
PAGAMENTO (Codigo, CodigoNegociacao, ValorParcela, Quitada, Parcelas, Desconto);
GERENTE_Realiza_NEGOCIACAO (CodigoGerente, CodigoNegociacao, Data);
PRODUTO (Codigo, Preco, Nome, Marca, Quantidade);
NEGOCIACAO_Contem_PRODUTO (CodigoNegociacao, CodigoProduto);
MEDIDAS (Codigo, Abdome, CoxaDireita, CoxaEsquerda, BracoDireito, BracoEsquerdo, Peitoral, AntebracoDireito, AntebracoEsquerdo, Deltoides, PanturrilhaEsquerda, PanturrilhaDireita);
PROFESSOR_ALUNO (CodigoProfessor, CodigoMedidas, CodigoAluno, Data);

3.2 Dicionário Lógico de Dados

ALUNO: Relação que armazena os dados de cada aluno da academia				
Atributo	**Descrição**	**Tipo**	**Domínio**	**Restrição**
Codigo	Atributo que representa o código de identificação do aluno	Int	Números inteiros positivos	▪ Chave substituta criada para representar o código de um aluno ▪ Chave Primária
Nome	Atributo que representa o nome do aluno	String (100)	String (100)	▪ Não Nulo
Sexo	Atributo que representa a inicial do sexo do aluno	Char	Char	▪ Não Nulo
Idade	Atributo que representa a idade do aluno	Int	Números inteiros positivos	▪ Não Nulo
Peso	Atributo que representa o peso do aluno	Real	Números reais positivos	▪ Não Nulo
Altura	Atributo que representa a altura do aluno	Real	Números reais positivos	▪ Não Nulo
BF	Atributo que representa o percentual do índice de gordura corporal do aluno	String (4)	String (4)	▪ Não Nulo

IMC	Atributo que representa o índice de massa corporal do aluno	Real	Números reais positivos	▪ Não Nulo
Telefone	Atributo que representa o número de telefone do aluno	String (20)	String (20)	▪ Não Nulo
EstadoSaude	Atributo que representa o estado de saúde em que o aluno se encontra	String (50)	String (50)	▪ Não Nulo
Objetivo	Atributo que representa o objetivo principal do aluno	String (100)	String (100)	▪ Não Nulo

Tabela 1: Relação ALUNO

MATRÍCULA: Relação que armazena os dados da matrícula de cada aluno da academia				
Atributo	**Descrição**	**Tipo**	**Domínio**	**Restrição**
Codigo	Atributo que representa o código da matrícula do aluno	Int	Números inteiros positivos	▪ Chave Primária
DataAbertura	Atributo que representa a data de realização da matrícula do aluno	Date	Date	▪ Não Nulo

DataTrancamento	Atributo que representa a data de trancamento de matrícula do aluno	Date	Date	▪ Sem Restrição
CodigoAluno	Atributo que armazena o código do aluno que realiza a matrícula	Int	Números inteiros positivos	▪ Não Nulo ▪ Único ▪ Chave estrangeira que referencia o atributo "Codigo" da relação "ALUNO"

Tabela 2: Relação MATRICULA

MENSALIDADE: Relação que armazena os dados da mensalidade de cada aluno da academia				
Atributo	**Descrição**	**Tipo**	**Domínio**	**Restrição**
Valor	Atributo que representa o valor da mensalidade paga pelo aluno	Real	Números reais positivos	▪ Não Nulo
DataRecibo	Atributo que representa a data de recebimento do valor correspondente à mensalidade do aluno	Date	Date	▪ Sem restrição
DataPagamento	Atributo que representa a data prevista para o pagamento da mensalidade do aluno	Date	Date	▪ Sem Restrição

CodigoAluno	Atributo que armazena o código do aluno que realiza a matrícula	Int	Números inteiros positivos	▪ Não Nulo ▪ Único ▪ Chave estrangeira que referencia o atributo “Codigo” da relação “ALUNO”
Codigo	Atributo que representa o código da mensalidade do aluno	Int	Números inteiros positivos	▪ Chave substituta criada para representar o código da mensalidade de um aluno ▪ Chave Primária

Tabela 3: Relação MENSALIDADE

TREINO: Relação que armazena os treinos de cada aluno da academia				
Atributo	**Descrição**	**Tipo**	**Domínio**	**Restrição**
Codigo	Atributo que representa o código do treino do aluno	Int	Números inteiros positivos	▪ Chave substituta criada para representar o código do treino de um aluno ▪ Chave Primária
Tipo	Atributo que representa o tipo de treino que será realizado pelo aluno	String (10)	String (10)	▪ Não Nulo
DescansoEntreSeries	Atributo que representa o tempo de descanso entre as series realizadas pelo aluno	String (5)	String (5)	▪ Não Nulo

UsoDeCarga	Atributo que representa a descrição do uso de carga do aluno	String (10)	String (10)	▪ Não Nulo
DescansoEntreRepeticoes	Atributo que representa o tempo de descanso entre as repetições realizadas pelo aluno	String (5)	String (5)	▪ Não Nulo
DescansoEntreCiclo	Atributo que representa a quantidade de dias de descanso entre os ciclos de exercícios realizados pelo aluno	Int	Int	▪ Não Nulo
Descricao	Atributo que representa uma breve descrição sobre os treinos realizados pelo aluno	String (200)	String (200)	▪ Sem Restrição

Tabela 4: Relação TREINO

ALUNO_Tem_TREINO: Relação que armazena os dados do relacionamento entre o aluno e o treino				
Atributo	**Descrição**	**Tipo**	**Domínio**	**Restrição**
CodigoAluno	Atributo que representa o código do aluno	Int	Números inteiros positivos	▪ Chave estrangeira que referencia o atributo “Codigo” da relação “ALUNO” ▪ Chave Primária

CodigoTreino	Atributo que representa o código do treino do aluno	Int	Números inteiros positivos	▪ Chave estrangeria que referencia o atributo “Codigo” da relação “TREINO” ▪ Chave Primária
Duracao	Atributo que representa a quantidade de meses que o treino do aluno vai durar	Int	Números inteiros positivos	▪ Não Nulo
DataInicio	Atributo que representa a data que o aluno iniciou os treinos	Date	Date	▪ Não Nulo

Tabela 5: Relação ALUNO_Tem_TREINO

EXERCICIO: Relação que armazena os dados de cada exercício do aluno				
Atributo	**Descrição**	**Tipo**	**Domínio**	**Restrição**
Codigo	Atributo que representa o código do exercício do Aluno	Int	Números inteiros positivos	▪ Chave substituta criada para representar o código do exercício de um aluno ▪ Chave Primária
Tipo	Atributo que representa o tipo de exercício que o aluno vai realizar	String (5)	String (5)	▪ Não Nulo

Series	Atributo que representa o número de series realizadas pelo aluno em um exercício	Int	Números inteiros positivos	▪ Não Nulo
Repeticoes	Atributo que armazena o número de repetições de uma série de exercícios	Int	Números inteiros positivos	▪ Não Nulo
Nome	Atributo que representa o nome do exercício	String (100)	String (100)	▪ Não Nulo
Musculo	Atributo que representa o nome do músculo trabalhado no exercício	String (50)	String (50)	▪ Não Nulo
DropSet	Atributo que armazena a forma de execução do exercício	Boolean	Boolean	▪ Não Nulo

Tabela 6: Relação EXERCICIO

TREINO_Tem_EXERCICIO: Relação que armazena os dados do relacionamento entre o treino e o exercício				
Atributo	**Descrição**	**Tipo**	**Domínio**	**Restrição**
CodigoTreino	Atributo que representa o código da relação treino	Int	Números inteiros positivos	▪ Chave estrangeira que referencia o atributo “Codigo” da relação “TREINO”

				▪ Chave Primária
CodigoExercicio	Atributo que representa o código da relação exercício	Int	Números inteiros positivos	▪ Chave estrangeira que referencia o atributo “Codigo” da relação “EXERCICIO” ▪ Chave Primária

Tabela 7: Relação TREINO_Tem_EXERCICIO

PROFESSOR: Relação que armazena os dados de cada professor da academia				
Atributo	**Descrição**	**Tipo**	**Domínio**	**Restrição**
CPF	Atributo que representa o CPF do professor	String (14)	String (14)	▪ Não Nulo ▪ Único
Sexo	Atributo que representa a sigla do sexo do professor	Char	Char	▪ Não Nulo
Nome	Atributo que representa o nome do professor	String (100)	String (100)	▪ Não Nulo
Codigo	Atributo que armazena o código do professor	Int	Int	▪ Chave substituta criada para representar o código do professor ▪ Chave Primária
DataPagamento	Atributo que representa a data referente ao pagamento do	Date	Date	▪ Não Nulo

	professor			
Salario	Atributo que representa o valor do salário pago ao professor	Real	Números reais positivos	▪ Não Nulo

Tabela 8: Relação PROFESSOR

EspecialidadePROFESSOR: Relação que armazena os dados das especialidades do professor da academia				
Atributo	**Descrição**	**Tipo**	**Domínio**	**Restrição**
CodigoProfessor	Atributo que armazena o código do professor	Int	Números inteiros positivos	▪ Chave estrangeira que referencia o atributo “Codigo” da relação “PROFESSOR” ▪ Chave Primária
Especialidade	Atributo que representa as especialidades do professor	String (100)	String (100)	▪ Chave Primária

Tabela 9: Relação EspecialidadePROFESSOR

PROFESSOR_EXERCICIO: Relação que armazena os dados do relacionamento entre o professor, o treino e o exercício				
Atributo	**Descrição**	**Tipo**	**Domínio**	**Restrição**
CodigoProfessor	Atributo que armazena o código do professor	Int	Números inteiros positivos	▪ Chave estrangeira que referencia o atributo “Codigo” da relação “PROFESSOR” ▪ Chave Primária

CodigoTreino	Atributo que armazena o código do treino	Int	Números inteiros positivos	▪ Chave estrangeira que referencia o atributo “Codigo” da relação “TREINO” ▪ Chave Primária
CodigoExercicio	Atributo que armazena o código do exercício	Int	Números inteiros positivos	▪ Chave estrangeira que referencia o atributo “Codigo” da relação “EXERCICIO” ▪ Chave Primária

Tabela 10: Relação PROFESSOR_EXERCICIO

GERENTE: Relação que armazena os dados de cada gerente da academia				
Atributo	**Descrição**	**Tipo**	**Domínio**	**Restrição**
CPF	Atributo que armazena o CPF do gerente da academia	String (14)	String (14)	▪ Não Nulo ▪ Único
Sexo	Atributo que representa a sigla do sexo do gerente da academia	Char	Char	▪ Não Nulo
Nome	Atributo que representa o nome do gerente da academia	String (100)	String (100)	▪ Não Nulo
Codigo	Atributo que representa o código do gerente da	Int	Números inteiros positivos	▪ Chave substituta criada para representar o

	academia			código do gerente ▪ Chave Primária
DataPagamento	Atributo que representa a data do pagamento do gerente	Date	Date	▪ Não Nulo

Tabela 11: Relação GERENTE

DESPESA: Relação que armazena os dados das despesas da academia				
Atributo	**Descrição**	**Tipo**	**Domínio**	**Restrição**
Descricao	Atributo que representa a descrição das despesas da academia	String (100)	String (100)	▪ Não Nulo
Valor	Atributo que representa o valor das despesas da academia	Real	Números reais positivos	▪ Não Nulo
Codigo	Atributo que armazena o código da despesa da academia	Int	Números inteiros positivos	▪ Chave substituta criada para representar o código da despesa da academia ▪ Chave Primária
Data	Atributo que armazena a data em que a despesa foi realizada	Date	Date	▪ Não Nulo

Tabela 12: Relação DESPESA

GERENTE_Gerencia_DESPESA: Relação que armazena os dados do relacionamento entre o gerente e a despesa				
Atributo	**Descrição**	**Tipo**	**Domínio**	**Restrição**
CodigoGerente	Atributo que armazena o código do gerente da academia	Int	Números inteiros positivos	▪ Chave estrangeira que referencia o atributo “Codigo” da relação “GERENTE” ▪ Chave Primária
CodigoDespesa	Atributo que armazena o código da despesa gerada pela academia	Int	Números inteiros positivos	▪ Chave estrangeira que referencia o atributo “Codigo” da relação “DESPESA” ▪ Chave Primária

Tabela 13: Relação GERENTE_Gerencia_DESPESA

NEGOCIACAO: Relação que armazena os dados de cada negociação realizada				
Atributo	**Descrição**	**Tipo**	**Domínio**	**Restrição**
Codigo	Atributo que armazena o código da negociação realizada	Int	Números inteiros positivos	▪ Chave substituta criada para representar o código da negociação ▪ Chave Primária
CNPJouCPF	Atributo que representa o CNPJ ou o CPF de quem realiza a negociação	String (20)	String (20)	▪ Não Nulo
Descricao	Atributo que representa a	String (100)	String (100)	▪ Não Nulo

	descrição de uma negociação realizada			
Valor	Atributo que armazena o valor de uma negociação realizada	Real	Números reais positivos	▪ Não Nulo
CodigoAluno	Atributo que armazena o código do aluno que fez a negociacao	Int	Números inteiros positivos	▪ Chave estrangeira que referencia o atributo “Codigo” da relação “ALUNO”
Data	Atributo que armazena a data em que a negociação foi feita pelo aluno	Date	Date	Date
Tipo	Atributo que armazena o tipo de negociação: compra/venda.	String (6)	String (6)	▪ Não nulo

Tabela 14: Relação NEGOCIACAO

PAGAMENTO: Relação que armazena os dados de pagamento				
Atributo	**Descrição**	**Tipo**	**Domínio**	**Restrição**
Codigo	Atributo que armazena o código do pagamento da negociação	Int	Números inteiros positivos	▪ Chave substituta criada para representar o código do pagamento ▪ Chave Primária

CodigoNegociacao	Atributo que armazena o código da negociação realizada	Int	Números inteiros positivos	▪ Não Nulo ▪ Único ▪ Chave estrangeira que referencia o atributo “Codigo” da relação “NEGOCIACAO”
ValorParcela	Atributo que armazena o valor de cada parcela da negociação	Real	Números reais positivos	▪ Não Nulo
Quitada	Atributo que representa se a negociação do tipo compra foi quitada ou não	Boolean	Boolean	▪ Não Nulo
Parcelas	Atributo que armazena a quantidade de parcelas de uma negociação do tipo compra	Int	Números inteiros positivos	▪ Não Nulo;
Desconto	Atributo que representa o valor do desconto da negociação do tipo compra caso seja à vista	Real	Números reais positivos	▪ Não Nulo

Tabela 15: Relação PAGAMENTO

GERENTE_Realiza_NEGOCIACAO: Relação que armazena os dados do relacionamento entre o gerente e a negociação				
Atributo	**Descrição**	**Tipo**	**Domínio**	**Restrição**

CodigoGerente	Atributo que armazena o código do gerente da academia	Int	Números inteiros positivos	▪ Chave estrangeira que referencia o atributo “Codigo” da relação “GERENTE” ▪ Chave Primária
CodigoNegociacao	Atributo que armazena o código da negociação realizada	Int	Números inteiros positivos	▪ Chave estrangeira que referencia o atributo “Codigo” da relação “NEGOCIACAO” ▪ Chave Primária
Data	Atributo que representa a data em que o gerente efetuou a negociação	Date	Date	▪ Não Nulo

Tabela 16: Relação GERENTE_Realiza_NEGOCIACAO

PRODUTO: Relação que armazena os dados de cada produto				
Atributo	**Descrição**	**Tipo**	**Domínio**	**Restrição**
Codigo	Atributo que armazena o código identificador de cada produto	Int	Números inteiros positivos	▪ Chave substituta criada para representar o código do produto ▪ Chave Primária
Preco	Atributo que armazena o valor do produto	Real	Números reais positivos	▪ Não Nulo
Nome	Atributo que representa o nome	String (100)	String (100)	▪ Não Nulo

	do produto			
Marca	Atributo que representa a marca do produto	String (50)	String (50)	▪ Não Nulo
Quantidade	Atributo que representa a quantidade de produtos feitos na negociação do tipo compra	Int	Números inteiros positivos	▪ Não Nulo

Tabela 17: Relação PRODUTO

NEGOCIACAO_Contem_PRODUTO: Relação que armazena os dados do relacionamento entre Negociacao e Produto				
Atributo	**Descrição**	**Tipo**	**Domínio**	**Restrição**
CodigoNegociacao	Atributo que armazena o código da negociação	Int	Números inteiros positivos	▪ Chave estrangeira que referencia o atributo “Codigo” da relação “NEGOCIACAO” ▪ Chave Primária
CodigoProduto	Atributo que armazena o código do produto	Int	Números inteiros positivos	▪ Chave estrangeira que referencia o atributo “Codigo” da relação “PRODUTO” ▪ Chave Primária

Tabela 18: Relação NEGOCIACAO_Contem_PRODUTO

MEDIDAS: Relação que armazena os dados de todas as medidas do aluno				
Atributo	**Descrição**	**Tipo**	**Domínio**	**Restrição**
Abdome	Atributo que representa as medidas em cm do abdome do aluno	Real	Números reais positivos	▪ Não Nulo
CoxaDireita	Atributo que representa as medidas em cm da coxa direita do aluno	Real	Números reais positivos	▪ Não Nulo
CoxaEsquerda	Atributo que representa as medidas em cm da coxa esquerda do aluno	Real	Números reais positivos	▪ Não Nulo
BracoDireito	Atributo que representa as medidas em cm do braço direito do aluno	Real	Números reais positivos	▪ Não Nulo
BracoEsquerdo	Atributo que representa as medidas em cm do braço esquerdo do aluno	Real	Números reais positivos	▪ Não Nulo
Peitoral	Atributo que representa as medidas em cm do peitoral do aluno	Real	Números reais positivos	▪ Não Nulo
AntebracoDireito	Atributo que representa as	Real	Números reais positivos	▪ Não Nulo

	medidas em cm do antebraço direito do aluno			
AntebracoEsquerdo	Atributo que representa as medidas em cm do antebraço esquerdo do aluno	Real	Números reais positivos	▪ Não Nulo
Deltoides	Atributo que representa as medidas em cm dos deltoides do aluno	Real	Números reais positivos	▪ Não Nulo
PanturrilhaEsquerda	Atributo que representa as medidas em cm da panturrilha esquerda do aluno	Real	Números reais positivos	▪ Não Nulo
PanturrilhaDireita	Atributo que representa as medidas em cm da panturrilha direita do aluno	Real	Números reais positivos	▪ Não Nulo
Codigo	Atributo que representa o código das medidas do aluno	Int	Números inteiros positivos	▪ Chave substituta criada para representar o código das medidas do aluno ▪ Chave Primária

Tabela 19: Relação MEDIDAS

PROFESSOR_ALUNO: Relação que armazena os dados do relacionamento entre Professor, Medidas e Aluno				
Atributo	**Descrição**	**Tipo**	**Domínio**	**Restrição**
CodigoProfessor	Atributo que armazena o código do professor	Int	Números inteiros positivos	▪ Chave estrangeira que referencia o atributo “Codigo” da relação “PROFESSOR” ▪ Chave Primária
CodigoAluno	Atributo que armazena o código do aluno	Int	Números inteiros positivos	▪ Chave estrangeira que referencia o atributo “Codigo” da relação “ALUNO” ▪ Chave Primária
CodigoMedidas	Atributo que armazena o código das medidas do aluno	Int	Números inteiros positivos	▪ Chave estrangeira que referencia o atributo “Codigo” da relação “MEDIDAS” ▪ Chave Primária
Data	Atributo que armazena a data em que as medidas foram tiradas	Date	Date	▪ Não Nulo

Tabela 20: Relação PROFESSOR_ALUNO

MODELAGEM FÍSICA

SISTEMA DE GERENCIAMENTO DE ACADEMIA FITNESS

4.1 Scripts SQL

4.1.1 Criação de Tabelas

Relação ALUNO

```
CREATE TABLE ALUNO(
        Codigo INT,
        Nome VARCHAR(100) NOT NULL,
        Sexo CHAR NOT NULL,
        Idade INT NOT NULL,
        Peso REAL NOT NULL,
        Altura REAL NOT NULL,
        BF VARCHAR(4) NOT NULL,
        IMC REAL NOT NULL,
        Telefone VARCHAR(20),
        EstadoSaude VARCHAR(50) NOT NULL,
        Objetivo VARCHAR(100) NOT NULL,
        CONSTRAINT  AlunoPK PRIMARY KEY (Codigo),
        CHECK(Peso>0),
        CHECK(Altura>0)
);
```

Relação MATRICULA

```
CREATE TABLE MATRICULA(
        Codigo INT,
        CodigoAluno INT NOT NULL UNIQUE,
        DataAbertura DATE NOT NULL,
        DataTrancamento DATE,
        CONSTRAINT  MatriculaPK PRIMARY KEY (Codigo),
        CONSTRAINT MatriculaFK FOREIGN KEY (CodigoAluno)
                REFERENCES ALUNO(Codigo)
```

```
        ON UPDATE CASCADE ON DELETE RESTRICT
);
```

Relação MENSALIDADE

```
CREATE TABLE MENSALIDADE(
        Codigo INT,
        CodigoAluno INT NOT NULL UNIQUE,
        Valor REAL NOT NULL,
        DataRecibo DATE,
        DataPagamento DATE,
        CONSTRAINT MensalidadePK PRIMARY KEY (Codigo),
        CONSTRAINT MensalidadeFK FOREIGN KEY (CodigoAluno)
                REFERENCES ALUNO(Codigo)
        ON UPDATE CASCADE ON DELETE RESTRICT,
        CHECK(Valor>0)
);
```

Relação TREINO

```
CREATE TABLE TREINO(
        Codigo INT,
        Tipo VARCHAR(10) NOT NULL,
        DescansoEntreSeries VARCHAR(5) NOT NULL,
        UsoDeCarga VARCHAR(10) NOT NULL,
        DescansoEntreRepeticoes VARCHAR(5) NOT NULL,
        DescansoEntreCiclo INT NOT NULL,
        Descricao VARCHAR(200),
        CONSTRAINT TreinoPK PRIMARY KEY (Codigo)
);
```

Relação ALUNO_Tem_TREINO

```
CREATE TABLE ALUNO_Tem_TREINO(
```

```
        CodigoAluno INT,
        CodigoTreino INT,
        Duracao INT NOT NULL,
        DataInicio DATE NOT NULL,
        CONSTRAINT Aluno_Tem_TreinoPK PRIMARY KEY (CodigoAluno,
        CodigoTreino),
        CONSTRAINT Aluno_Tem_TreinoFK FOREIGN KEY (CodigoAluno)
                REFERENCES ALUNO(Codigo)
        ON UPDATE CASCADE ON DELETE RESTRICT,
        CONSTRAINT Aluno_Tem_TreinoFK2 FOREIGN KEY (Codigotreino)
                REFERENCES TREINO(Codigo)
        ON UPDATE CASCADE ON DELETE RESTRICT
);
```

Relação EXERCICIO

```
CREATE TABLE EXERCICIO(
        Codigo INT,
        Tipo VARCHAR(5) NOT NULL,
        Series INT NOT NULL,
        Repeticoes INT NOT NULL,
        Nome VARCHAR(100) NOT NULL,
        Musculo VARCHAR(50) NOT NULL,
        DropSet BOOLEAN NOT NULL,
        CONSTRAINT ExercicioPK PRIMARY KEY (Codigo),
        CHECK(Series>0),
        CHECK(Repeticoes>6)
);
```

Relação TREINO_Tem_EXERCICIO

```
CREATE TABLE TREINO_Tem_EXERCICIO(
        CodigoTreino INT,
```

```
    CodigoExercicio INT,
    CONSTRAINT Treino_Tem_ExercicioPK PRIMARY KEY (CodigoTreino,
    CodigoExercicio),
    CONSTRAINT Treino_Tem_ExercicioFK FOREIGN KEY (CodigoTreino)
        REFERENCES TREINO(Codigo)
    ON UPDATE CASCADE ON DELETE RESTRICT,
    CONSTRAINT Treino_Tem_ExercicioFK2 FOREIGN KEY (CodigoExercicio)
        REFERENCES EXERCICIO(Codigo)
    ON UPDATE CASCADE ON DELETE RESTRICT
);
```

Relação PROFESSOR

```
CREATE TABLE PROFESSOR(
    CPF VARCHAR(14) NOT NULL UNIQUE,
    Sexo CHAR NOT NULL,
    Nome VARCHAR(100) NOT NULL,
    Codigo INT,
    DataPagamento DATE NOT NULL,
    Salario REAL NOT NULL,
    CONSTRAINT ProfessorPK PRIMARY KEY (Codigo),
    CHECK(Salario>0)
);
```

Relação EspecialidadePROFESSOR

```
CREATE TABLE EspecialidadePROFESSOR(
    CodigoProfessor INT,
    Especialidade VARCHAR(100),
    CONSTRAINT EspecialidadeProfessorPK PRIMARY KEY (CodigoProfessor,
    Especialidade),
    CONSTRAINT EspecialidadeProfessorFK FOREIGN KEY (CodigoProfessor)
        REFERENCES PROFESSOR(Codigo)
```

```
        ON UPDATE CASCADE ON DELETE RESTRICT
);
```

Relação PROFESSOR_EXERCICIO

```
CREATE TABLE PROFESSOR_EXERCICIO(
        CodigoProfessor INT,
        CodigoTreino INT,
        CodigoExercicio INT,
        CONSTRAINT Professor_ExercicioPK PRIMARY KEY (CodigoProfessor,
        CodigoTreino, CodigoExercicio),
        CONSTRAINT Professor_ExercicioFK FOREIGN KEY (CodigoProfessor)
                REFERENCES PROFESSOR(Codigo)
        ON UPDATE CASCADE ON DELETE RESTRICT,
        CONSTRAINT Professor_ExercicioFK2 FOREIGN KEY (CodigoTreino)
                REFERENCES TREINO(Codigo)
        ON UPDATE CASCADE ON DELETE RESTRICT,
        CONSTRAINT Professor_ExercicioFK3 FOREIGN KEY (CodigoExercicio)
                REFERENCES EXERCICIO(Codigo)
        ON UPDATE CASCADE ON DELETE RESTRICT
);
```

Relação GERENTE

```
CREATE TABLE GERENTE(
        Codigo INT,
        CPF VARCHAR(14) NOT NULL UNIQUE,
        Sexo CHAR NOT NULL,
        Nome VARCHAR(100) NOT NULL,
        DataPagamento DATE NOT NULL,
        CONSTRAINT GerentePK PRIMARY KEY (Codigo)
);
```

Relação DESPESA

```
CREATE TABLE DESPESA(
        Codigo INT,
        Descricao VARCHAR(100) NOT NULL,
        Valor REAL NOT NULL,
        Data DATE NOT NULL,
        CONSTRAINT DespesaPK PRIMARY KEY (Codigo)
);
```

Relação GERENTE_Gerencia_DESPESA

```
CREATE TABLE GERENTE_Gerencia_DESPESA(
        CodigoGerente INT,
        CodigoDespesa INT,
        CONSTRAINT GERENTE_Gerencia_DESPESAPK PRIMARY KEY
        (CodigoGerente, CodigoDespesa),
        CONSTRAINT Gerente_Gerencia_DespesaFK FOREIGN KEY (CodigoGerente)
                REFERENCES GERENTE(Codigo)
        ON UPDATE CASCADE ON DELETE RESTRICT,
        CONSTRAINT Gerente_Gerencia_DespesaFK2 FOREIGN KEY
        (CodigoDespesa)
                REFERENCES DESPESA(Codigo)
        ON UPDATE CASCADE ON DELETE RESTRICT
);
```

Relação NEGOCIACAO

```
CREATE TABLE NEGOCIACAO(
        Codigo INT,
        CodigoAluno INT,
        CNPJouCPF VARCHAR(20) NOT NULL,
        Descricao VARCHAR(100) NOT NULL,
        Valor REAL NOT NULL,
```

```
        Data DATE,
        Tipo VARCHAR(6) NOT NULL,
        CONSTRAINT NegociacaoPK PRIMARY KEY (Codigo),
        CONSTRAINT NegociacaoFK FOREIGN KEY (CodigoAluno)
                REFERENCES ALUNO(Codigo)
        ON UPDATE CASCADE ON DELETE RESTRICT,
        CHECK(Valor>0)
);
```

Relação PAGAMENTO

```
CREATE TABLE PAGAMENTO(
        Codigo INT,
        CodigoNegociacao INT NOT NULL UNIQUE,
        ValorParcela REAL NOT NULL,
        Quitada BOOLEAN NOT NULL,
        Parcelas INT NOT NULL,
        Desconto REAL NOT NULL,
        CONSTRAINT CompraAprazoPK PRIMARY KEY (Codigo),
        CONSTRAINT PagamentoFK FOREIGN KEY (CodigoNegociacao)
                REFERENCES NEGOCIACAO(Codigo)
        ON UPDATE CASCADE ON DELETE RESTRICT,
        CHECK(ValorParcela>0),
        CHECK(Parcelas>0 AND Parcelas<=10)
);
```

Relação GERENTE_Realiza_NEGOCIACAO

```
CREATE TABLE GERENTE_Realiza_NEGOCIACAO(
        CodigoGerente INT,
        CodigoNegociacao INT,
        Data DATE NOT NULL,
        CONSTRAINT Gerente_Realiza_NegociacaoPK PRIMARY KEY (CodigoGerente,
```

```
        CodigoNegociacao),
        CONSTRAINT Gerente_Realiza_NegociacaoFK FOREIGN KEY (CodigoGerente)
                REFERENCES GERENTE(Codigo)
        ON UPDATE CASCADE ON DELETE RESTRICT,
        CONSTRAINT Gerente_Realiza_NegociacaoFK2 FOREIGN KEY
        (CodigoNegociacao)
                REFERENCES NEGOCIACAO(Codigo)
        ON UPDATE CASCADE ON DELETE RESTRICT
);
```

Relação PRODUTO

```
CREATE TABLE PRODUTO(
        Codigo INT,
        Preco REAL NOT NULL,
        Nome VARCHAR(100) NOT NULL,
        Marca VARCHAR(50) NOT NULL,
        Quantidade INT NOT NULL,
        CONSTRAINT PRODUTOPK PRIMARY KEY (Codigo),
        CHECK(Preco>0),
        CHECK(Quantidade>0)
);
```

Relação NEGOCIACAO_Contem_PRODUTO

```
CREATE TABLE NEGOCIACAO_Contem_PRODUTO(
        CodigoNegociacao INT,
        CodigoProduto INT,
        CONSTRAINT Negociacao_Contem_ProdutoPK PRIMARY KEY
        (CodigoNegociacao, CodigoProduto),
        CONSTRAINT Negociacao_Contem_ProdutoFK FOREIGN KEY
        (CodigoNegociacao)
                REFERENCES NEGOCIACAO(Codigo)
```

```
        ON UPDATE CASCADE ON DELETE RESTRICT,
        CONSTRAINT Negociacao_Contem_ProdutoFK2 FOREIGN KEY
        (CodigoProduto)
            REFERENCES Produto(Codigo)
        ON UPDATE CASCADE ON DELETE RESTRICT
);
```

Relação MEDIDAS

```
CREATE TABLE MEDIDAS(
        Codigo INT,
        Abdome REAL NOT NULL,
        CoxaDireita REAL NOT NULL,
        CoxaEsquerda REAL NOT NULL,
        BracoDireito REAL NOT NULL,
        BracoEsquerdo REAL NOT NULL,
        Peitoral REAL NOT NULL,
        AntebracoDireito REAL NOT NULL,
        AntebracoEsquerdo REAL NOT NULL,
        Deltoides REAL NOT NULL,
        PanturrilhaEsquerda REAL NOT NULL,
        PanturrilhaDireita REAL NOT NULL,
        CONSTRAINT MEDIDASPK PRIMARY KEY (Codigo)
);
```

Relação PROFESSOR_ALUNO

```
CREATE TABLE PROFESSOR_ALUNO(
        CodigoProfessor INT,
        CodigoMedidas INT,
        CodigoAluno INT,
        Data DATE NOT NULL,
        CONSTRAINT Professor_AlunoPK PRIMARY KEY (CodigoProfessor,
```

```
        CodigoAluno, CodigoMedidas),
        CONSTRAINT Professor_AlunoFK FOREIGN KEY (CodigoProfessor)
                REFERENCES PROFESSOR(Codigo)
        ON UPDATE CASCADE ON DELETE RESTRICT,
        CONSTRAINT Professor_AlunoFK2 FOREIGN KEY (CodigoAluno)
                REFERENCES ALUNO(Codigo)
        ON UPDATE CASCADE ON DELETE RESTRICT,
        CONSTRAINT Professor_AlunoFK3 FOREIGN KEY (CodigoMedidas)
                REFERENCES MEDIDAS(Codigo)
        ON UPDATE CASCADE ON DELETE RESTRICT
);
```

4.1.2 Inserções

Relação ALUNO

INSERT INTO ALUNO (Codigo, Nome, Sexo, Idade, Peso, Altura, BF, IMC, Telefone, EstadoSaude, Objetivo) VALUES(11111, 'João de Sousa', 'M', 20, 80.8, 1.70, '20%', 27.7, '(83) 99834-0672', 'Acima do Peso', 'Emagrecimento');

INSERT INTO ALUNO(Codigo, Nome, Sexo, Idade, Peso, Altura, BF, IMC, Telefone, EstadoSaude, Objetivo) VALUES(11112, 'Pedro da Silva', 'M', 25, 75.3, 1.65, '18%', 27.5, '(83) 99939-4097', 'Acima do Peso', 'Emagrecimento');

INSERT INTO ALUNO(Codigo, Nome, Sexo, Idade, Peso, Altura, BF, IMC, Telefone, EstadoSaude, Objetivo) VALUES(11113, 'Maria do Carmo Pereira', 'F', 19, 58.7, 1.62, '15%', 22.1, '(83) 99834-0672', 'Peso Normal', 'Hipertrofia Muscular');

INSERT INTO ALUNO(Codigo, Nome, Sexo, Idade, Peso, Altura, BF, IMC, Telefone, EstadoSaude, Objetivo) VALUES(11114, 'Francisco Carlos de Araújo', 'M', 28, 85.4, 1.74, '26%', 28.1, '(83) 99476-9835', 'Acima do Peso', 'Emagrecimento');

INSERT INTO ALUNO(Codigo, Nome, Sexo, Idade, Peso, Altura, BF, IMC, Telefone, EstadoSaude, Objetivo) VALUES(11115, 'Joana Pereira de Sousa', 'F', 18, 56.2, 1.65, '12%', 20.6, '(83) 99855-4568', 'Peso Normal', 'Hipertrofia Muscular');

Relação MATRICULA

INSERT INTO MATRICULA(Codigo, CodigoAluno, DataAbertura, DataTrancamento) VALUES(22222, 11111, '02/01/2017', '02/03/2017');

INSERT INTO MATRICULA(Codigo, CodigoAluno, DataAbertura, DataTrancamento) VALUES(22223, 11112, '05/01/2017', null);

INSERT INTO MATRICULA(Codigo, CodigoAluno, DataAbertura, DataTrancamento) VALUES(22224, 11113, '07/01/2017', null);

INSERT INTO MATRICULA(Codigo, CodigoAluno, DataAbertura, DataTrancamento) VALUES(22225, 11114, '10/01/2017', null);

INSERT INTO MATRICULA(Codigo, CodigoAluno, DataAbertura, DataTrancamento) VALUES(22226, 11115, '15/01/2017', null);

Relação MENSALIDADE

INSERT INTO MENSALIDADE(Codigo, CodigoAluno, Valor, DataRecibo, DataPagamento) VALUES(33333, 11111, 50, '15/02/2017', '02/02/2017');
INSERT INTO MENSALIDADE(Codigo, CodigoAluno, Valor, DataRecibo, DataPagamento) VALUES(33334, 11112, 50, '05/03/2017', '05/03/2017');
INSERT INTO MENSALIDADE(Codigo, CodigoAluno, Valor, DataRecibo, DataPagamento) VALUES(33335, 11113, 50, '09/03/2017', '07/03/2017');
INSERT INTO MENSALIDADE(Codigo, CodigoAluno, Valor, DataRecibo, DataPagamento) VALUES(33336, 11114, 50, '18/03/2017', '10/03/2017');
INSERT INTO MENSALIDADE(Codigo, CodigoAluno, Valor, DataRecibo, DataPagamento) VALUES(33337, 11115, 50, '15/03/2017', '15/03/2017');

Relação TREINO

INSERT INTO TREINO(Codigo, Tipo, DescansoEntreSeries, UsoDeCarga, DescansoEntreRepeticoes, DescansoEntreCiclo, Descricao) VALUES(44444, 'A', '30s', 'Moderada', '1m', 2, 'Perda de Peso');
INSERT INTO TREINO(Codigo, Tipo, DescansoEntreSeries, UsoDeCarga, DescansoEntreRepeticoes, DescansoEntreCiclo, Descricao) VALUES(44445, 'B', '30s', 'Leve', '1m', 1, 'Perda de Peso');
INSERT INTO TREINO(Codigo, Tipo, DescansoEntreSeries, UsoDeCarga, DescansoEntreRepeticoes, DescansoEntreCiclo, Descricao) VALUES(44446, 'C', '30s', 'Pesada', '1m', 3, 'Ganho de Massa Muscular');
INSERT INTO TREINO(Codigo, Tipo, DescansoEntreSeries, UsoDeCarga, DescansoEntreRepeticoes, DescansoEntreCiclo, Descricao) VALUES(44447, 'D', '30s', 'Pesada', '1m', 3, 'Ganho de Massa Muscular');
INSERT INTO TREINO(Codigo, Tipo, DescansoEntreSeries, UsoDeCarga, DescansoEntreRepeticoes, DescansoEntreCiclo, Descricao) VALUES(44448, 'E', '30s', 'Moderada', '1m', 2, 'Perda de Peso');
INSERT INTO TREINO(Codigo, Tipo, DescansoEntreSeries, UsoDeCarga, DescansoEntreRepeticoes, DescansoEntreCiclo, Descricao) VALUES(44449, 'F', '30s', 'Moderada', '1m', 2, 'Perda de Peso');

Relação ALUNO_Tem_TREINO

INSERT INTO ALUNO_Tem_TREINO(CodigoAluno, CodigoTreino, Duracao, DataInicio) VALUES(11111, 44444, 2, '02/02/2017');

INSERT INTO ALUNO_Tem_TREINO(CodigoAluno, CodigoTreino, Duracao, DataInicio) VALUES(11112, 44445, 2, '05/03/2017');

INSERT INTO ALUNO_Tem_TREINO(CodigoAluno, CodigoTreino, Duracao, DataInicio) VALUES(11113, 44446, 2, '07/04/2017');

INSERT INTO ALUNO_Tem_TREINO(CodigoAluno, CodigoTreino, Duracao, DataInicio) VALUES(11114, 44447, 2, '10/03/2017');

INSERT INTO ALUNO_Tem_TREINO(CodigoAluno, CodigoTreino, Duracao, DataInicio) VALUES(11115, 44448, 2, '15/03/2017');

Relação EXERCICIO

INSERT INTO EXERCICIO(Codigo, Tipo, Series, Repeticoes, Nome, Musculo, DropSet) VALUES(55555, 'A', 3, 12, 'Supino Declinado com Halteres', 'Peitoral', true);

INSERT INTO EXERCICIO(Codigo, Tipo, Series, Repeticoes, Nome, Musculo, DropSet) VALUES(55556, 'A', 3, 12, 'Crucifixo', 'Peitoral', false);

INSERT INTO EXERCICIO(Codigo, Tipo, Series, Repeticoes, Nome, Musculo, DropSet) VALUES(55557, 'A', 3, 12, 'Pec Deck', 'Peitoral', true);

INSERT INTO EXERCICIO(Codigo, Tipo, Series, Repeticoes, Nome, Musculo, DropSet) VALUES(55558, 'B', 3, 12, 'Puxada Atrás com Polia Alta', 'Dorsais', true);

INSERT INTO EXERCICIO(Codigo, Tipo, Series, Repeticoes, Nome, Musculo, DropSet) VALUES(55559, 'B', 3, 12, 'Remada com Halter', 'Dorsais', true);

INSERT INTO EXERCICIO(Codigo, Tipo, Series, Repeticoes, Nome, Musculo, DropSet) VALUES(55510, 'B', 3, 12, 'Pillover na Polia Alta com Barra', 'Dorsais', true);

INSERT INTO EXERCICIO(Codigo, Tipo, Series, Repeticoes, Nome, Musculo, DropSet) VALUES(55511, 'C', 3, 12, 'Agachamento com Barra', 'Pernas', false);

INSERT INTO EXERCICIO(Codigo, Tipo, Series, Repeticoes, Nome, Musculo, DropSet) VALUES(55512, 'C', 3, 12, 'Bom Dia', 'Pernas', false);

INSERT INTO EXERCICIO(Codigo, Tipo, Series, Repeticoes, Nome, Musculo, DropSet)

VALUES(55513, 'C', 3, 12, 'Máquina Adutora', 'Pernas', false);
INSERT INTO EXERCICIO(Codigo, Tipo, Series, Repeticoes, Nome, Musculo, DropSet)
VALUES(55514, 'D', 3, 12, 'Remada Alta com Barra', 'Ombros', true);
INSERT INTO EXERCICIO(Codigo, Tipo, Series, Repeticoes, Nome, Musculo, DropSet)
VALUES(55515, 'D', 3, 12, 'Press Militar com Barra à Nuca', 'Ombros', true);
INSERT INTO EXERCICIO(Codigo, Tipo, Series, Repeticoes, Nome, Musculo, DropSet)
VALUES(55516, 'D', 3, 12, 'Elevações Frontais com Halteres', 'Ombros', true);
INSERT INTO EXERCICIO(Codigo, Tipo, Series, Repeticoes, Nome, Musculo, DropSet)
VALUES(55517, 'E', 3, 12, 'Testa Inclinado', 'Bíceps, Tríceps e Antebraços', false);
INSERT INTO EXERCICIO(Codigo, Tipo, Series, Repeticoes, Nome, Musculo, DropSet)
VALUES(55518, 'E', 3, 12, 'Rosca Direta Barra', 'Bíceps, Tríceps e Antebraços', true);
INSERT INTO EXERCICIO(Codigo, Tipo, Series, Repeticoes, Nome, Musculo, DropSet)
VALUES(55519, 'E', 3, 12, 'Rosca Pronada', 'Bíceps, Tríceps e Antebraços', false);
INSERT INTO EXERCICIO(Codigo, Tipo, Series, Repeticoes, Nome, Musculo, DropSet)
VALUES(55520, 'F', 3, 12, 'Supra Abdominal', 'Peitoral, Ombros', true);
INSERT INTO EXERCICIO(Codigo, Tipo, Series, Repeticoes, Nome, Musculo, DropSet)
VALUES(55521, 'F', 3, 12, 'Infra Abdominal', 'Abdome', false);
INSERT INTO EXERCICIO(Codigo, Tipo, Series, Repeticoes, Nome, Musculo, DropSet)
VALUES(55522, 'F', 3, 12, 'Elevações de Pernas Tipo Burro', 'Pernas', false);

Relação TREINO_Tem_EXERCICIO

INSERT INTO TREINO_Tem_EXERCICIO(CodigoTreino, CodigoExercicio)
VALUES(44444, 55555);
INSERT INTO TREINO_Tem_EXERCICIO(CodigoTreino, CodigoExercicio)
VALUES(44444, 55556);
INSERT INTO TREINO_Tem_EXERCICIO(CodigoTreino, CodigoExercicio)
VALUES(44444, 55557);
INSERT INTO TREINO_Tem_EXERCICIO(CodigoTreino, CodigoExercicio)
VALUES(44445, 55558);
INSERT INTO TREINO_Tem_EXERCICIO(CodigoTreino, CodigoExercicio)
VALUES(44445, 55559);

INSERT INTO TREINO_Tem_EXERCICIO(CodigoTreino, CodigoExercicio) VALUES(44445, 55510);
INSERT INTO TREINO_Tem_EXERCICIO(CodigoTreino, CodigoExercicio) VALUES(44446, 55511);
INSERT INTO TREINO_Tem_EXERCICIO(CodigoTreino, CodigoExercicio) VALUES(44446, 55512);
INSERT INTO TREINO_Tem_EXERCICIO(CodigoTreino, CodigoExercicio) VALUES(44446, 55513);
INSERT INTO TREINO_Tem_EXERCICIO(CodigoTreino, CodigoExercicio) VALUES(44447, 55514);
INSERT INTO TREINO_Tem_EXERCICIO(CodigoTreino, CodigoExercicio) VALUES(44447, 55515);
INSERT INTO TREINO_Tem_EXERCICIO(CodigoTreino, CodigoExercicio) VALUES(44447, 55516);
INSERT INTO TREINO_Tem_EXERCICIO(CodigoTreino, CodigoExercicio) VALUES(44448, 55517);
INSERT INTO TREINO_Tem_EXERCICIO(CodigoTreino, CodigoExercicio) VALUES(44448, 55518);
INSERT INTO TREINO_Tem_EXERCICIO(CodigoTreino, CodigoExercicio) VALUES(44448, 55519);
INSERT INTO TREINO_Tem_EXERCICIO(CodigoTreino, CodigoExercicio) VALUES(44449, 55520);
INSERT INTO TREINO_Tem_EXERCICIO(CodigoTreino, CodigoExercicio) VALUES(44449, 55521);
INSERT INTO TREINO_Tem_EXERCICIO(CodigoTreino, CodigoExercicio) VALUES(44449, 55522);

Relação PROFESSOR

INSERT INTO PROFESSOR(CPF, Sexo, Nome, Codigo, DataPagamento, Salario) VALUES('111.111.111-01', 'M', 'José Ricardo Oliveira', 66666, '01/04/2017', 1000);
INSERT INTO PROFESSOR(CPF, Sexo, Nome, Codigo, DataPagamento, Salario)

VALUES('222.222.333-02', 'M', 'André de Sousa Dias', 66667, '02/04/2017', 1000);
INSERT INTO PROFESSOR(CPF, Sexo, Nome, Codigo, DataPagamento, Salario) VALUES('333.333.333-03', 'M', 'Marcos Rogerio da Costa', 66668, '03/04/2017', 1000);
INSERT INTO PROFESSOR(CPF, Sexo, Nome, Codigo, DataPagamento, Salario) VALUES('444.444.444-04', 'M', 'Alexandre da Silva Cartaxo', 66669, '04/04/2017', 1000);
INSERT INTO PROFESSOR(CPF, Sexo, Nome, Codigo, DataPagamento, Salario) VALUES('555.555.555-05', 'F', 'Andréia Rodrigues de Oliveira', 66610, '05/04/2017', 1500);

Relação EspecialidadePROFESSOR

INSERT INTO EspecialidadePROFESSOR(CodigoProfessor, Especialidade) VALUES(66666, 'Educador Físico');
INSERT INTO EspecialidadePROFESSOR(CodigoProfessor, Especialidade) VALUES(66667, 'Educador Físico');
INSERT INTO EspecialidadePROFESSOR(CodigoProfessor, Especialidade) VALUES(66668, 'Educador Físico');
INSERT INTO EspecialidadePROFESSOR(CodigoProfessor, Especialidade) VALUES(66669, 'Educador Físico');
INSERT INTO EspecialidadePROFESSOR(CodigoProfessor, Especialidade) VALUES(66610, 'Personal Trainer');

Relação PROFESSOR_EXERCICIO

INSERT INTO PROFESSOR_EXERCICIO(CodigoProfessor, CodigoTreino, CodigoExercicio) VALUES(66666, 44444, 55555);
INSERT INTO PROFESSOR_EXERCICIO(CodigoProfessor, CodigoTreino, CodigoExercicio) VALUES(66666, 44444, 55556);
INSERT INTO PROFESSOR_EXERCICIO(CodigoProfessor, CodigoTreino, CodigoExercicio) VALUES(66666, 44444, 55557);
INSERT INTO PROFESSOR_EXERCICIO(CodigoProfessor, CodigoTreino, CodigoExercicio) VALUES(66667, 44445, 55558);
INSERT INTO PROFESSOR_EXERCICIO(CodigoProfessor, CodigoTreino,

CodigoExercicio) VALUES(66667, 44445, 55559);
INSERT INTO PROFESSOR_EXERCICIO(CodigoProfessor, CodigoTreino, CodigoExercicio) VALUES(66667, 44445, 55510);
INSERT INTO PROFESSOR_EXERCICIO(CodigoProfessor, CodigoTreino, CodigoExercicio) VALUES(66668, 44446, 55511);
INSERT INTO PROFESSOR_EXERCICIO(CodigoProfessor, CodigoTreino, CodigoExercicio) VALUES(66668, 44446, 55512);
INSERT INTO PROFESSOR_EXERCICIO(CodigoProfessor, CodigoTreino, CodigoExercicio) VALUES(66668, 44446, 55513);
INSERT INTO PROFESSOR_EXERCICIO(CodigoProfessor, CodigoTreino, CodigoExercicio) VALUES(66669, 44447, 55514);
INSERT INTO PROFESSOR_EXERCICIO(CodigoProfessor, CodigoTreino, CodigoExercicio) VALUES(66669, 44447, 55515);
INSERT INTO PROFESSOR_EXERCICIO(CodigoProfessor, CodigoTreino, CodigoExercicio) VALUES(66669, 44447, 55516);
INSERT INTO PROFESSOR_EXERCICIO(CodigoProfessor, CodigoTreino, CodigoExercicio) VALUES(66610, 44448, 55517);
INSERT INTO PROFESSOR_EXERCICIO(CodigoProfessor, CodigoTreino, CodigoExercicio) VALUES(66610, 44448, 55518);
INSERT INTO PROFESSOR_EXERCICIO(CodigoProfessor, CodigoTreino, CodigoExercicio) VALUES(66610, 44448, 55519);
INSERT INTO PROFESSOR_EXERCICIO(CodigoProfessor, CodigoTreino, CodigoExercicio) VALUES(66610, 44449, 55520);
INSERT INTO PROFESSOR_EXERCICIO(CodigoProfessor, CodigoTreino, CodigoExercicio) VALUES(66610, 44449, 55521);
INSERT INTO PROFESSOR_EXERCICIO(CodigoProfessor, CodigoTreino, CodigoExercicio) VALUES(66610, 44449, 55522);

Relação GERENTE

INSERT INTO GERENTE(Codigo, CPF, Sexo, Nome, DataPagamento) VALUES(77777, '666.666.666-06', 'M', 'Felipe Augusto Vilela', '01/04/2017');

INSERT INTO GERENTE(Codigo, CPF, Sexo, Nome, DataPagamento) VALUES(77778, '777.777.777-07', 'M', 'Sérgio Brito Ferrari', '02/04/2017');
INSERT INTO GERENTE(Codigo, CPF, Sexo, Nome, DataPagamento) VALUES(77779, '888.888.888-08', 'F', 'Valesca Maria da Costa', '03/04/2017');
INSERT INTO GERENTE(Codigo, CPF, Sexo, Nome, DataPagamento) VALUES(77710, '999.999.999-09', 'M', 'Cristovão de Lima', '04/04/2017');
INSERT INTO GERENTE(Codigo, CPF, Sexo, Nome, DataPagamento) VALUES(77711, '101.101.101-10', 'M', 'Fernando Pereira Belém', '05/04/2017');

Relação DESPESA

INSERT INTO DESPESA(Codigo, Descricao, Valor, Data) VALUES(88888, 'Água, Energia', 500, '01/04/2017');
INSERT INTO DESPESA(Codigo, Descricao, Valor, Data) VALUES(88889, 'Água, Energia', 600, '02/04/2017');
INSERT INTO DESPESA(Codigo, Descricao, Valor, Data) VALUES(88810, 'Água, Energia', 400, '03/04/2017');
INSERT INTO DESPESA(Codigo, Descricao, Valor, Data) VALUES(88811, 'Água, Energia', 700, '04/04/2017');
INSERT INTO DESPESA(Codigo, Descricao, Valor, Data) VALUES(88812, 'Água, Energia', 200, '05/04/2017');

Relação GERENTE_Gerencia_DESPESA

INSERT INTO GERENTE_Gerencia_DESPESA(CodigoGerente, CodigoDespesa) VALUES(77777, 88888);
INSERT INTO GERENTE_Gerencia_DESPESA(CodigoGerente, CodigoDespesa) VALUES(77778, 88889);
INSERT INTO GERENTE_Gerencia_DESPESA(CodigoGerente, CodigoDespesa) VALUES(77779, 88810);
INSERT INTO GERENTE_Gerencia_DESPESA(CodigoGerente, CodigoDespesa) VALUES(77710, 88811);
INSERT INTO GERENTE_Gerencia_DESPESA(CodigoGerente, CodigoDespesa)

VALUES(77711, 88812);

Relação NEGOCIACAO

INSERT INTO NEGOCIACAO(Codigo, CodigoAluno, CNPJouCPF, Descricao, Valor, Data, Tipo) VALUES(99999, 11111, '666.666.666-06', 'Hipertróficos', 900, '01/03/2017', 'Venda');

INSERT INTO NEGOCIACAO(Codigo, CodigoAluno, CNPJouCPF, Descricao, Valor, Data, Tipo) VALUES(99913, null, '11.111.111/1111-11', 'Pastas Proteicas', 95.04, null, 'Compra');

INSERT INTO NEGOCIACAO(Codigo, CodigoAluno, CNPJouCPF, Descricao, Valor, Data, Tipo) VALUES(99914, null, '11.111.111/1111-11', 'Hipertróficos', 1800, null, 'Compra');

INSERT INTO NEGOCIACAO(Codigo, CodigoAluno, CNPJouCPF, Descricao, Valor, Data, Tipo) VALUES(99915, null, '22.222.222/2222-22', 'Barras Cereais', 590, null, 'Compra');

INSERT INTO NEGOCIACAO(Codigo, CodigoAluno, CNPJouCPF, Descricao, Valor, Data, Tipo) VALUES(99916, null, '33.333.333/3333-33', 'Energéticos', 1500, null, 'Compra');

INSERT INTO NEGOCIACAO(Codigo, CodigoAluno, CNPJouCPF, Descricao, Valor, Data, Tipo) VALUES(99917, null, '44.444.444/4444-44', 'Termogênicos', 3000, null, 'Compra');

INSERT INTO NEGOCIACAO(Codigo, CodigoAluno, CNPJouCPF, Descricao, Valor, Data, Tipo) VALUES(99918, 11111, '666.666.666-06', 'Termogênicos', 300, '01/03/2017', 'Venda');

Relação PAGAMENTO

INSERT INTO PAGAMENTO(Codigo, CodigoNegociacao, ValorParcela, Quitada, Parcelas, Desconto) VALUES(54321, 99999, 100, false, 9, 0);

INSERT INTO PAGAMENTO(Codigo, CodigoNegociacao, ValorParcela, Quitada, Parcelas, Desconto) VALUES(98765, 99918, 50, false, 6, 0);

INSERT INTO PAGAMENTO(Codigo, CodigoNegociacao, ValorParcela, Quitada,

Parcelas, Desconto) VALUES(15192, 99917, 600, false, 5, 0);
INSERT INTO PAGAMENTO(Codigo, CodigoNegociacao, ValorParcela, Quitada, Parcelas, Desconto) VALUES(87932, 99916, 500, false, 3, 0);
INSERT INTO PAGAMENTO(Codigo, CodigoNegociacao, ValorParcela, Quitada, Parcelas, Desconto) VALUES(97621, 99915, 295, false, 2, 0);
INSERT INTO PAGAMENTO(Codigo, CodigoNegociacao, ValorParcela, Quitada, Parcelas, Desconto) VALUES(97622, 99914, 360, false, 5, 0);
INSERT INTO PAGAMENTO(Codigo, CodigoNegociacao, ValorParcela, Quitada, Parcelas, Desconto) VALUES(97623, 99913, 95.04, true, 1, 10.56);

Relação GERENTE_Realiza_NEGOCIACAO

INSERT INTO GERENTE_Realiza_NEGOCIACAO(CodigoGerente, CodigoNegociacao, Data) VALUES(77778, 99914, '02/03/2017');
INSERT INTO GERENTE_Realiza_NEGOCIACAO(CodigoGerente, CodigoNegociacao, Data) VALUES(77779, 99916, '05/03/2017');
INSERT INTO GERENTE_Realiza_NEGOCIACAO(CodigoGerente, CodigoNegociacao, Data) VALUES(77710, 99917, '08/03/2017');
INSERT INTO GERENTE_Realiza_NEGOCIACAO(CodigoGerente, CodigoNegociacao, Data) VALUES(77711, 99915, '01/03/2017');
INSERT INTO GERENTE_Realiza_NEGOCIACAO(CodigoGerente, CodigoNegociacao, Data) VALUES(77711, 99914, '01/03/2017');
INSERT INTO GERENTE_Realiza_NEGOCIACAO(CodigoGerente, CodigoNegociacao, Data) VALUES(77779, 99913, '01/03/2017');

Relação PRODUTO

INSERT INTO PRODUTO(Codigo, Preco, Nome, Marca, Quantidade) VALUES(10101, 450, 'Whey Protein', 'Optimum Nutrition', 2);
INSERT INTO PRODUTO(Codigo, Preco, Nome, Marca, Quantidade) VALUES(20202, 300, 'Fat Destroyer', 'Atlhetica Nutrition', 1);
INSERT INTO PRODUTO(Codigo, Preco, Nome, Marca, Quantidade) VALUES(30303, 150, 'Fat Destroyer', 'Atlhetica Nutrition', 15);

INSERT INTO PRODUTO(Codigo, Preco, Nome, Marca, Quantidade) VALUES(40404, 150, 'Turbo Force', 'Uni Flora', 10);
INSERT INTO PRODUTO(Codigo, Preco, Nome, Marca, Quantidade) VALUES(50505, 118, 'Bar Ultimate Fire Black', 'Max Titanium', 5);
INSERT INTO PRODUTO(Codigo, Preco, Nome, Marca, Quantidade) VALUES(60606, 300, 'Whey Protein', 'Max Titanium', 6);
INSERT INTO PRODUTO(Codigo, Preco, Nome, Marca, Quantidade) VALUES(70707, 8.80, 'Pasta Amendoin 33g proteina', 'Body Action', 12);

Relação NEGOCIACAO_Contem_PRODUTO

INSERT INTO NEGOCIACAO_Contem_PRODUTO(CodigoNegociacao, CodigoProduto) VALUES(99999, 10101);
INSERT INTO NEGOCIACAO_Contem_PRODUTO(CodigoNegociacao, CodigoProduto) VALUES(99918, 20202);
INSERT INTO NEGOCIACAO_Contem_PRODUTO(CodigoNegociacao, CodigoProduto) VALUES(99917, 30303);
INSERT INTO NEGOCIACAO_Contem_PRODUTO(CodigoNegociacao, CodigoProduto) VALUES(99916, 40404);
INSERT INTO NEGOCIACAO_Contem_PRODUTO(CodigoNegociacao, CodigoProduto) VALUES(99915, 50505);
INSERT INTO NEGOCIACAO_Contem_PRODUTO(CodigoNegociacao, CodigoProduto) VALUES(99914, 60606);
INSERT INTO NEGOCIACAO_Contem_PRODUTO(CodigoNegociacao, CodigoProduto) VALUES(99913, 70707);

Relação MEDIDAS

INSERT INTO MEDIDAS(Codigo, Abdome, CoxaDireita, CoxaEsquerda, BracoDireito, BracoEsquerdo, Peitoral, AntebracoDireito, AntebracoEsquerdo, Deltoides, PanturrilhaEsquerda, PanturrilhaDireita) VALUES(12345, 90, 54, 53, 36, 38, 100, 25, 26, 50, 40, 38);
INSERT INTO MEDIDAS(Codigo, Abdome, CoxaDireita, CoxaEsquerda, BracoDireito,

BracoEsquerdo, Peitoral, AntebracoDireito, AntebracoEsquerdo, Deltoides, PanturrilhaEsquerda, PanturrilhaDireita) VALUES(67891, 85, 52, 50, 37, 36, 97, 24, 23, 60, 39, 38);

INSERT INTO MEDIDAS(Codigo, Abdome, CoxaDireita, CoxaEsquerda, BracoDireito, BracoEsquerdo, Peitoral, AntebracoDireito, AntebracoEsquerdo, Deltoides, PanturrilhaEsquerda, PanturrilhaDireita) VALUES(10111, 60, 52, 51, 24, 25, 88, 22, 23, 40, 36, 35);

INSERT INTO MEDIDAS (Codigo, Abdome, CoxaDireita, CoxaEsquerda, BracoDireito, BracoEsquerdo, Peitoral, AntebracoDireito, AntebracoEsquerdo, Deltoides, PanturrilhaEsquerda, PanturrilhaDireita) VALUES (12131, 95, 55, 54, 39, 38, 102, 29, 28, 52, 33, 34);

INSERT INTO MEDIDAS (Codigo, Abdome, CoxaDireita, CoxaEsquerda, BracoDireito, BracoEsquerdo, Peitoral, AntebracoDireito, AntebracoEsquerdo, Deltoides, PanturrilhaEsquerda, PanturrilhaDireita) VALUES (14151, 62, 51, 52, 23, 24, 86, 24, 25, 43, 32, 31);

Relação PROFESSOR_ALUNO

INSERT INTO PROFESSOR_ALUNO (CodigoProfessor, CodigoMedidas, CodigoAluno, Data) VALUES (66666, 12345, 11111, '01/04/2017');

INSERT INTO PROFESSOR_ALUNO (CodigoProfessor, CodigoMedidas, CodigoAluno, Data) VALUES (66667, 67891, 11112, '02/04/2017');

INSERT INTO PROFESSOR_ALUNO (CodigoProfessor, CodigoMedidas, CodigoAluno, Data) VALUES (66668, 10111, 11113, '03/04/2017');

INSERT INTO PROFESSOR_ALUNO (CodigoProfessor, CodigoMedidas, CodigoAluno, Data) VALUES (66669, 12131, 11114, '04/04/2017');

INSERT INTO PROFESSOR_ALUNO (CodigoProfessor, CodigoMedidas, CodigoAluno, Data) VALUES (66610, 14151, 11115, '05/04/2017');

4.1.3 Consultas

Dispostas abaixo o levantamento e resoluções:

1. Todas as negociações que aluno João de Sousa fez. O resultado deve conter o código e o nome do aluno, o codigo da compra, a descrição, o valor e a data da referente.

   ```
   SELECT A.Nome, N.CodigoAluno, N.Codigo, N.Descricao, N.Valor, N.Data
   FROM Negociacao N, Aluno A
   WHERE A.Codigo = N.CodigoAluno AND
   A.Nome = 'João de Sousa'
   ```

2. Todas as negociações realizadas por cada gerente, ordenando-os por sexo e nome, selecionando o código, nome e sexo, e o código, a descrição, o valor e a data da compra referente.

   ```
   SELECT GRN.CodigoGerente, G.Nome, G.Sexo, GRN.CodigoNegociacao,
   N.Descricao, N.Valor, GRN.Data
   FROM Gerente_Realiza_Negociacao GRN, Negociacao N, Gerente G
   Where G.Codigo = GRN.CodigoGerente AND
   N.Codigo = GRN.CodigoNegociacao
   ORDER BY G.Sexo, G.Nome
   ```

3. Selecione o código e nome do aluno, junto ao código, descrição, valor e data da negociação que estão pendentes de pagamento pelo aluno João de Sousa.

   ```
   SELECT N.CodigoAluno, A.Nome, N.Codigo AS CodigoNegociacao,
   N.Descricao, N.Valor, N.Data
   FROM ALUNO A, Negociacao N JOIN Pagamento P ON N.Codigo =
   P.CodigoNegociacao AND P.Quitada = false
   WHERE A.Codigo = N.CodigoAluno AND
   A.Nome = 'João de Sousa'
   ```

4. O código, nome e sexo do gerente, junto ao código, descrição, valor e data da negociação que estão pendentes de pagamento pelo gerente de nome Valesca Maria da Costa.

```
SELECT G.Codigo AS Gerente, G.Nome, G.Sexo, N.Codigo AS
CodigoNegociacao, N.Descricao, N.Valor, GRN.Data
FROM GERENTE G, NEGOCIACAO N, PAGAMENTO P,
GERENTE_Realiza_NEGOCIACAO GRN
WHERE G.Codigo = GRN.CodigoGerente AND
GRN.CodigoNegociacao = N.Codigo AND
P.CodigoNegociacao = GRN.CodigoNegociacao AND
P.Quitada = false AND
GRN.CodigoGerente IN
(SELECT Codigo
FROM GERENTE
WHERE Nome = 'Valesca Maria da Costa')
```

5. Todos os dados de produto, com o código da negociação, a descrição e o valor, junto ao código, nome, e sexo do aluno João de Sousa.

```
SELECT A.Codigo AS Aluno, A.Nome, A.Sexo, N.Codigo AS Negociacao,
N.Descricao, N.Valor, P.Codigo AS Produto, P.Nome, P.Preco, P.Marca,
P.Quantidade
FROM Negociacao N, Aluno A, Produto P, Negociacao_Contem_Produto NCP
WHERE A.Nome = 'João de Sousa' AND
N.Codigo = NCP.CodigoNegociacao AND
P.Codigo = NCP.CodigoProduto AND
N.CodigoAluno = A.Codigo
```

6. O código e o nome de todos os alunos, junto ao valor, data de pagamento previsto e a data da realização do pagamento de fato da mensalidade.

```
SELECT A.Codigo, A.Nome, M.Valor, M.DataPagamento AS
DataPagamentoPrevista, M.DataRecibo AS DataRealizacaoPagamento
```

```
FROM ALUNO A, MENSALIDADE M
WHERE A.Codigo = M.CodigoAluno
```

7. Os dados das compras, com o nome de todos os referidos alunos, mesmo que o aluno ainda não tenha feito compra.

```
SELECT A.Codigo, A.Nome, N.CNPJouCPF,  N.Codigo, N.Descricao, N.Valor,
N.Data
FROM (ALUNO A LEFT OUTER JOIN NEGOCIACAO N ON A.Codigo =
N.CodigoAluno)
```

8. Informações de medidas, por uma determinada data: 2017-04-05, do aluno Joana Pereira de Sousa.

```
SELECT *
FROM MEDIDAS M
WHERE M.Codigo IN
(SELECT PA.CodigoMedidas
 FROM PROFESSOR_ALUNO PA
 WHERE PA.Data = '2017-04-05' AND
 PA.CodigoAluno IN
(SELECT Codigo
 FROM ALUNO
 WHERE Nome = 'Joana Pereira de Sousa'))
```

9. As características dos exercícios do treino do aluno João de Sousa. A consulta deverá conter o nome do aluno procurado, o tipo de treino que está treinando, a data que começou a treinar e o tempo que o treino deve durar, a descrição do treino e as informações dos exercícios de cada dia (de cada siga do tipo).

```
SELECT A.Nome, T.Tipo AS TipoTreino, ATT.DataInicio, ATT.Duracao,
T.Descricao, E.Codigo AS CodigoExercicio, E.Tipo, E.Series, E.Repeticoes,
E.Nome, E.Musculo, E.DropSet
FROM ALUNO A, TREINO T, ALUNO_Tem_TREINO ATT, EXERCICIO E,
TREINO_Tem_EXERCICIO TTE
```

```
WHERE ATT.CodigoTreino = T.Codigo AND
TTE.CodigoExercicio = E.Codigo AND
TTE.CodigoTreino = ATT.CodigoTreino AND
ATT.CodigoAluno = A.Codigo AND
A.Codigo IN
(SELECT Codigo
 FROM ALUNO
 WHERE Nome = 'João de Sousa')
```

10. O nome e a matricula de todos os alunos que adquiriram em alguma negociação o produto Whey Protein, não importando a marca.

```
(SELECT A.Nome, M.Codigo AS Matricula
FROM ALUNO A, MATRICULA M, NEGOCIACAO N, PRODUTO P,
NEGOCIACAO_CONTEM_PRODUTO NCP
WHERE P.Nome = 'Whey Protein' AND
A.Codigo = N.CodigoAluno AND
P.Codigo = NCP.CodigoProduto AND
N.Codigo = NCP.CodigoNegociacao AND
A.Codigo = M.CodigoAluno)
INTERSECT
(SELECT A.Nome, M.Codigo AS MAtricula
 FROM ALUNO A, MATRICULA M)
```

11. O salário, o nome e o sexo dos professores ordenados pelo sexo e nome.

```
SELECT Nome, Sexo, Salario
FROM PROFESSOR
ORDER BY Sexo, Nome
```

12. A quantidade de alunos ativos na academia no momento da consulta do sistema.

```
SELECT COUNT(*) NumeroAlunosAtivos
FROM MATRICULA
WHERE DataTrancamento IS NULL OR
```

```
DataTrancamento > (SELECT CURRENT_DATE)
```

13. Os códigos e o nomes dos alunos, dispostos em ordem alfabética ascendente, agrupados por código, nome do aluno e do produto, com todos os nomes dos produtos já negociados por ele.

```
(SELECT A.Codigo, A.Nome AS NomeAluno, P.Nome
FROM ALUNO A, PRODUTO P, NEGOCIACAO N,
Negociacao_Contem_Produto NCP
WHERE A.Codigo = N.CodigoAluno AND
P.Codigo = NCP.CodigoProduto AND
N.Codigo = NCP.CodigoNegociacao
GROUP BY A.Codigo, A.Nome, P.Nome)
INTERSECT
(SELECT A.Codigo, A.Nome, P.Nome
 FROM ALUNO A, PRODUTO P)
ORDER BY NomeAluno
```

14. Qual tipo de treino e o professor que montou o treino de Pedro da Silva. A consulta deve retornar o nome do professor, o nome do aluno, o tipo de treino de Pedro e a descrição do tipo de treino.

```
SELECT P.Nome AS Professor, A.Nome AS Aluno, T.Tipo AS TipoTreino,
T.Descricao
FROM ALUNO A, TREINO T, ALUNO_Tem_TREINO ATT, PROFESSOR P,
PROFESSOR_EXERCICIO PE
WHERE ATT.CodigoTreino = PE.CodigoTreino AND
PE.CodigoProfessor = P.Codigo AND
ATT.CodigoAluno = A.Codigo AND
T.Codigo = ATT.CodigoTreino AND
A.Codigo IN
(SELECT Codigo
 FROM ALUNO
 WHERE Nome = 'Pedro da Silva')
```

15. Os alunos que ainda não pagaram a mensalidade de um determinado mês.

```
SELECT A.Nome
FROM ALUNO A, MENSALIDADE M
WHERE A.Codigo = M.CodigoAluno AND M.DataPagamento IS NULL
```

16. Nome de todos os alunos que tem o primeiro nome começando com a letra “J” ou “R”.

```
SELECT A.Nome
FROM ALUNO A
WHERE A.Nome LIKE 'J%' OR A.Nome LIKE 'R%'
```

17. O nome dos professores que tenham o primeiro nome começando com “Andr”, que sejam do sexo feminino e ganham mais de R$ 1.000,00.

```
SELECT P.Nome
FROM PROFESSOR P
WHERE P.Nome LIKE 'Andr%' AND P.Sexo = 'F' AND P.Salario > 1000
```

18. Nome dos alunos que fizeram alguma negociação e ainda não pagaram totalmente.

```
SELECT A.Nome
FROM ALUNO A
WHERE EXISTS
(SELECT * FROM NEGOCIACAO N
WHERE A.Codigo = N.CodigoAluno AND N.Tipo = 'Venda' AND NOT EXISTS
(SELECT * FROM PAGAMENTO P
WHERE N.Codigo = P.CodigoNegociacao AND P.Quitada = true))
```

19. Os gerentes que fizeram alguma negociação e ainda não pagaram totalmente, eliminando resultados repetidos.

```
SELECT DISTINCT G.Nome
FROM GERENTE G, GERENTE_Realiza_NEGOCIACAO GRN
WHERE EXISTS
```

```
(SELECT * FROM NEGOCIACAO N
WHERE G.Codigo = GRN.CodigoGerente AND N.Codigo =
GRN.CodigoNegociacao AND N.Tipo = 'Compra' AND NOT EXISTS
(SELECT * FROM PAGAMENTO P
WHERE GRN.CodigoNegociacao = P.CodigoNegociacao AND P.Quitada = true))
```

20. A quantidade de alunos do sexo masculino agrupados por grupo.

```
SELECT COUNT(*) AS QuantidadeAlunosSexoMasculino
FROM ALUNO A
WHERE A.Sexo = 'M'
GROUP BY A.Sexo
```

4.1.4 Visões

Dispostas abaixo o levantamento e resoluções:

1. O código,nome, sexo, idade, telefone e data de trancamento dos alunos que estão ativos na academia no momento da consulta do sistema.

   ```
   CREATE VIEW AlunosAtivos AS
   SELECT A.Codigo, A.Nome, A.Sexo, A.Idade, A.Telefone, M.DataTrancamento
   FROM ALUNO A, MATRICULA M
   WHERE A.Codigo = M.CodigoAluno AND
   M.DataTrancamento IS NULL OR
   M.DataTrancamento > (SELECT CURRENT_DATE)
   ORDER BY A.Nome
   ```

2. O CPF, nome e sexo de todos os professores ordenados por sexo da academia.

   ```
   CREATE VIEW ProfessoresDaAcademia AS
   SELECT CPF, Nome, Sexo
   FROM PROFESSOR
   ORDER BY Sexo DESC
   ```

4.1.5 Índices

Dispostos abaixo:

```
CREATE INDEX AlunoIndice ON ALUNO(Codigo);
CREATE INDEX MatriculaIndice ON MATRICULA(Codigo);
CREATE INDEX ProdutoIndice ON PRODUTO(Codigo);
CREATE INDEX NegociacaoIndice ON NEGOCIACAO(Codigo);
CREATE INDEX PagamentoIndice ON PAGAMENTO(Codigo);
CREATE INDEX MensalidadeIndice ON MENSALIDADE(Codigo);
CREATE INDEX TreinoIndice ON TREINO(Codigo);
CREATE INDEX ExercicioIndice ON EXERCICIO(Codigo);
CREATE INDEX ProfessorIndice ON PROFESSOR(Codigo);
CREATE INDEX GerenteIndice ON GERENTE(Codigo);
CREATE INDEX DespesaIndice ON DESPESA(Codigo);
CREATE INDEX MedidasIndice ON MEDIDAS(Codigo);
```

4.1.6 Procedimentos Armazenados

Dispostos abaixo o levantamento e resoluções:

1. Procedimento que calcule o BF (*body fat*) de todos os alunos.

```
CREATE OR REPLACE FUNCTION BF(INT)
RETURNS REAL AS '
DECLARE
        aluno ALUNO%ROWTYPE;
        matricula ALIAS FOR $1;
BEGIN
        SELECT INTO aluno * FROM ALUNO
        WHERE Codigo = matricula;
        IF(aluno.Sexo = "M") THEN
                RETURN ((1.20 * IMC(aluno.Codigo)) + (0.23 * aluno.Idade) - (10.8
                * 1) - 5.4);
        END IF;
        RETURN ((1.20 * IMC(aluno.Codigo)) + (0.23 * aluno.Idade) - (10.8 * 0) -
        5.4);
END
' LANGUAGE plpgsql;
```

2. Procedimento que calcule o IMC de todos os alunos.

```
CREATE OR REPLACE FUNCTION IMC(INT)
RETURNS REAL AS '
DECLARE
        aluno ALUNO%ROWTYPE;
        matricula ALIAS FOR $1;
BEGIN
        SELECT INTO aluno * FROM ALUNO
        WHERE Codigo = matricula;
```

```
        RETURN aluno.peso/(aluno.altura*aluno.altura);
END
' LANGUAGE plpgsql;
```