CAMPUS CAJAZEIRAS

Análise e Desenvolvimento de Sistemas

Banco de Dados I

Flávio Túlio Fabrício de Almeida

Guilherme Henrique de Abreu Pessoa

Karlos Messyas da Silva Paulino

Samuel Andrade de Araújo

PROJETO DE BANCO DE DADOS: DIÁRIO ESCOLAR

Cajazeiras - PB

2023.2

Flávio Túlio Fabrício de Almeida
Guilherme Henrique de Abreu Pessoa
Karlos Messyas da Silva Paulino
Samuel Andrade de Araújo

PROJETO DE BANCO DE DADOS: DIÁRIO ESCOLAR

Trabalho apresentado ao Curso Superior Tecnológico de Análise e Desenvolvimento de Sistemas do IFPB – Instituto Federal de Educação, Ciência e Tecnologia da Paraíba Campus Cajazeiras, para a disciplina de Banco de Dados I, ministrado pelo Prof. Dr. Fabio Gomes de Andrade.

Cajazeiras - PB
2023.2

Sumário

Etapa 1 - Conceitual

1.1 - Introdução

O sistema de diário escolar desempenha um papel fundamental na gestão e acompanhamento do desempenho acadêmico dos alunos, bem como na organização das atividades escolares. Para garantir a eficiência e a qualidade desse processo, é essencial contar com um banco de dados bem estruturado e robusto que atenda às necessidades específicas das instituições de ensino. Este projeto conceitual de banco de dados visa oferecer uma abordagem abrangente e eficaz para o desenvolvimento de um sistema de diário escolar moderno e funcional.

Ao longo deste projeto, exploraremos os principais requisitos e desafios envolvidos na criação de um banco de dados que suporte a gestão de informações acadêmicas, como registros de alunos, notas, frequência, calendário escolar e outras informações relevantes para o ambiente escolar. Além disso, abordaremos a importância da segurança, escalabilidade e usabilidade na concepção desse sistema, garantindo que ele atenda às necessidades tanto dos educadores quanto dos alunos e seus responsáveis.

Pretendemos criar um projeto conceitual flexível, adaptável e de fácil manutenção, para isso, exploraremos os conceitos fundamentais de modelagem de dados, integridade referencial e normalização.

À medida que avançamos neste projeto, buscaremos identificar os requisitos específicos do domínio da aplicação em questão e adaptar a estrutura do banco de dados para atender a essas necessidades particulares.

1.2 - Descrição do domínio de aplicação

O sistema escolar deve possuir os dados dos funcionários, armazenando seu nome, matrícula, CPF, data de admissão, salário, e-mail, e até dois números de celular, além dos dados do endereço. Entre os funcionários, há professores e secretários.

O secretário é o responsável por matricular alunos, além de receber as justificativas de faltas dos alunos. Para justificar a falta será necessário um documento comprobatório e as datas, todo secretário possui um cargo e horário de trabalho definido para o mesmo.

O professor possui dados da sua área de atuação e sua titulação, onde todo professor é responsável por ministrar pelo menos uma disciplina. A disciplina guarda informações sobre o total de aulas e sua carga horária, além de ter um identificador.

As disciplinas possuem aulas e turmas, onde cada aula tem um identificador, a data e o conteúdo da aula e cada turma armazena os dados da quantidade de alunos, o local (sala), o ano letivo e a série de cada turma. Além disso, devem ser registradas as

informações referentes a cada registro de aulas, incluindo a turma e a disciplina, além de sua data e conteúdo.

O sistema também deve armazenar dados sobre os alunos, como seu nome, matrícula, CPF, sexo, data de nascimento, telefone dos responsáveis, nome do responsável (guardião que efetuou a matrícula) e seu endereço (estado, município, CEP, bairro, logradouro, número e zona, caso seja urbana ou rural). Além disso, deve armazenar o ano em que o aluno está matriculado.

Todo aluno participa de uma turma e realiza avaliações das disciplinas contempladas pela turma. Cada avaliação se faz necessário ter os dados do tipo da avaliação (prova, trabalho, projeto, recuperação), a data e a nota. O aluno também está associado aos bimestres, para saber seu desempenho em cada um e sua situação (aprovado, reprovado ou cursando), todo bimestre armazena seu número de referência e o ano letivo do qual faz parte, além da data de início e término de cada bimestre. Todo bimestre possui avaliações.

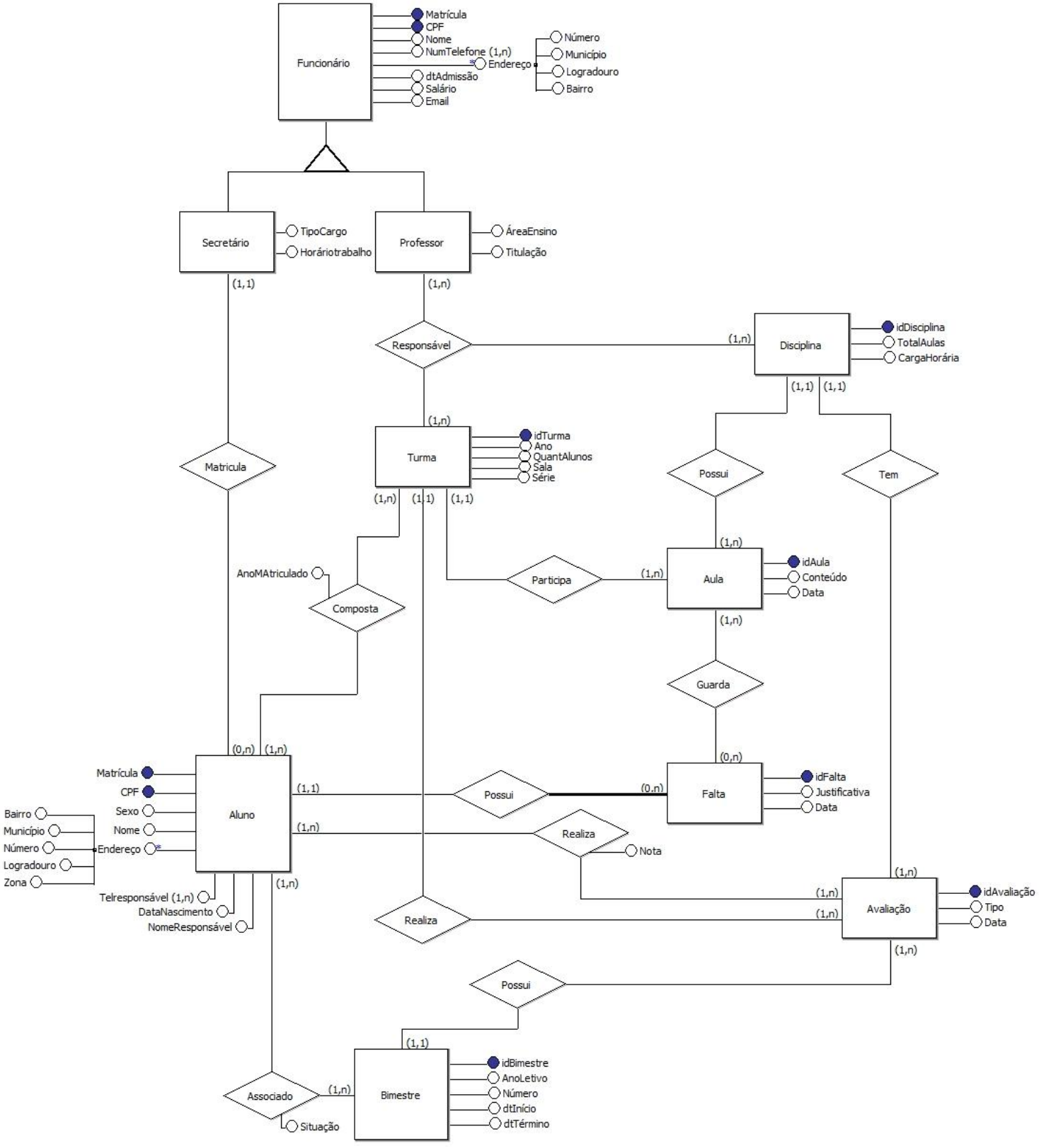
1.3 - Consultas

Para atender as demandas propostas, o banco de dados deve atender inicialmente as seguintes consultas:

- Quais alunos foram matriculados em cada turma;
- Total de faltas de cada aluno (as presenças serão abstraídas de acordo com o total de aulas);
- A nota de cada aluno em cada avaliação;
- A média das notas de cada aluno em cada bimestre;
- O contato do(s) responsável(eis) de cada aluno;
- A localização de cada aluno;
- A média final de cada aluno no ano letivo;
- Quais professores ministram determinada disciplina;
- Quais aulas de determinada disciplina foram ministradas para determinada turma;
- O nível de formação de cada professor;

1.4 - Diagrama entidade-relacionamento

Dado o domínio descrito anteriormente e as consultas que devem ser atendidas pelo banco de dados, se segue o diagrama entidade-relacionamento (doravante denominado pela sigla DER) que descreve o domínio da aplicação.


Funcionário
Matrícula
CPF
Nome
NumTelefone (1,n)
Endereço*
Número
Município
Logradouro
Bairro
dtAdmissão
Salário
Email
Secretário
TipoCargo
Horáriotrabalho
Professor
ÁreaEnsino
Titulação
(1,1)
(1,n)
Responsável
(1,n)
Disciplina
idDisciplina
TotalAulas
CargaHorária
(1,1)
(1,1)
(1,n)
Turma
idTurma
Ano
QuantAlunos
Sala
Série
Matricula
Possui
Tem
(1,n)
(1,n)
(1,1)
(1,1)
(1,n)
AnoMAtriculado
Composta
Participa
(1,n)
Aula
idAula
Conteúdo
Data
(1,n)
Guarda
(0,n)
(1,n)
Matrícula
CPF
Sexo
Nome
Endereço*
Bairro
Município
Número
Logradouro
Zona
Aluno
(1,1)
(1,n)
Possui
(0,n)
Falta
idFalta
Justificativa
Data
Realiza
Nota
Telresponsável (1,n)
DataNascimento
NomeResponsável
(1,n)
(1,n)
(1,n)
Realiza
(1,n)
Avaliação
idAvaliação
Tipo
Data
(1,n)
Possui
(1,1)
Associado
(1,n)
Bimestre
idBimestre
AnoLetivo
Número
dtInício
dtTérmino
Situação

1.5 - Dicionário de dados conceitual

A seguinte descrição descreve cada entidade e cada atributo presente no diagrama entidade-relacionamento para melhor entendimento do domínio aqui apresentado e para o planejamento do banco de dados.

1.5.1 - Entidades

Entidade Funcionário:

- ➢ Generalização das entidades Secretário e Professor, tem como função armazenar dados em comum de um funcionário;

Atributo(s):

- Matrícula: armazena a matrícula do funcionário;
- CPF: armazena o CPF do funcionário;
- Nome: Armazena o nome do funcionário;
- NumTelefone: Armazena o(s) número(s) de telefone do funcionário;
- Endereço: Armazena o endereço do funcionário:
  - Logradouro: armazena o logradouro onde reside o aluno;
  - Número: armazena o número da casa do aluno;
  - Bairro: armazena o nome do bairro em que o aluno mora;
  - Município: armazena o município em que o aluno mora;
- dtAdmissão: armazena a data em que o funcionário foi admitido;
- Salário: armazena o salário do funcionário;
- Email: Armazena o email do funcionário;

Entidade Secretário:

- ➢ Especialização da entidade Funcionário, herda todos os atributos da entidade mais geral, e é utilizada para armazenar as informações de um secretário;

Atributo(s):

- TipoCargo: armazena o cargo específico do funcionário;
- HorárioTrabalho: armazena o horário de trabalho do secretário;

Entidade Professor

- ➢ Especialização da entidade Funcionário, herda todos os atributos da entidade mais geral, e é utilizada para armazenar as informações de um professor;

Atributo(s):

- ÁreaEnsino: armazena a área em que o professor atua;
- Titulação: armazena a titulação do professor;

Entidade Aluno:

- ➢ Entidade responsável por armazenar os dados de um aluno;

Atributo(s):

- Matrícula: armazena a matrícula do aluno;
- CPF: armazena o CPF do aluno;
- Nome: armazena o nome do aluno;

- Sexo: armazena o sexo do aluno;
- Endereço: armazena as informações do endereço do aluno:
    - Logradouro: armazena o logradouro onde reside o aluno;
    - Número: armazena o número da casa do aluno;
    - Bairro: armazena o nome do bairro em que o aluno mora;
    - Município: armazena o município em que o aluno mora;
    - Zona: armazena a zona em que o aluno reside;
- TelResponsavel: armazena o(s) número(s) de telefone dos responsáveis;
- DataNasc: armazena a data de nascimento do aluno;
- NomeResponsavel: armazena o nome do responsável pelo aluno;

Entidade Bimestre:

➢ Entidade que armazena todas as informações de todos os bimestres letivos;

Atributo(s):

- idBimestre: armazena o identificador único do bimestre;
- AnoLetivo: armazena o ano letivo do bimestre em questão;
- Número: armazena o número referente do bimestre;
- dtInício: armazena a data de início do bimestre;
- dtTérmino: armazena a data em que o bimestre terminou;

Entidade Disciplina:

➢ Entidade que armazena as informações sobre as disciplinas lecionadas;

Atributo(s):

- idDisciplina; armazena o identificador único da disciplina;
- TotalAulas: armazena a quantidade de aulas semanais da disciplina;
- CargaHorária: armazena a carga horária anual da disciplina;

Entidade Avaliação:

➢ Entidade que armazena as informações sobre todas as avaliações, não os resultados de notas dos alunos mas sim as avaliações que foram aplicadas;

Atributo(s):

- idAvaliação: armazena o identificador único da avaliação;
- Tipo: armazena o tipo da avaliação;
- Data: armazena a data em que o aluno realizou a avaliação;

Entidade Turma:

➢ Entidade que armazena as informações sobre as turmas existentes;

Atributo(s):

- idTurma: armazena o identificador único para cada turma;
- Ano: armazena o ano em questão da turma;
- QuantAlunos: armazena a quantidade de alunos da turma;
- Sala: armazena o número da sala em que a turma estuda;

- Série: armazena a série em que a turma está;

Entidade Aula:

➢ Entidade que armazena todas as informações sobre as aulas;

Atributo(s):

- idAula: armazena o identificador único da aula;
- Conteúdo: armazena o conteúdo ministrado na aula;
- Data: armazena a data em que a aula aconteceu;

Entidade Falta:

➢ Entidade fraca da entidade Aluno que armazena as informações sobre as faltas dos alunos;

Atributo(s):

- idFalta: armazena o identificador único da falta;
- Justificativa: armazena a justificativa da falta;
- Data: armazena a data da falta;

1.5.2 - Relacionamentos

- *Secretário* **Matrícula** *Aluno*: descreve as matrículas anuais de cada aluno por um secretário, onde cada aluno é matriculado por um secretário apenas e um secretário pode matricular muitos ou nenhum aluno;
- *Professor* é **Responsável** por *Disciplina* para *Turma*: descreve quais professores ministram as disciplinas para as turmas, cada professor pode ministrar diversas disciplinas para diversas turmas, cada turma pode ter diversos professores ministrando diversas disciplinas, e cada disciplina pode ser ministrada por diversos professores para diversas turmas;
- *Turma* é **Composta** por *Aluno*: descreve os alunos que compõem cada turma, onde cada aluno pode estar em mais de uma turma (pois um aluno pode passar de ano, podendo então ser registrado em mais de uma turma no banco caso informações de turmas anteriores sejam necessárias), cada turma pode ser composta por muitos alunos. Possui o atributo AnoMatriculado, que se refere ao ano em que o aluno foi matriculado nesta turma;
- *Turma* **Participa** de *Aula*: descreve as aulas que determinada turma participou, uma turma pode participar de diversas aulas mas cada aula é ministrada para apenas uma turma;
- *Disciplina* **Possui** *Aula*: registra as disciplinas referentes a cada aula, sendo que cada disciplina pode possuir diversas aulas e cada aula é associada a apenas uma disciplina;
- *Aluno* **Possui** *Falta*: registra as faltas de cada aluno individualmente para depois decidir, com base na justificativa, se a falta será ou não deduzida do total de aulas, registra a falta com relação ao dia inteiro, não cada aula individualmente. cada falta será associada a um único aluno e um aluno pode ter várias faltas;

- *Aula* **Guarda** *Falta*: registra as aulas a que cada falta será associada, como uma falta registra um dia inteiro, esse relacionamento irá registrar as aulas associadas a esta falta, cada falta é associada a diversas aulas e cada aula pode ter diversas faltas;
- *Aluno* **Realiza** *Avaliação*: registra as avaliações aplicadas para cada aluno, a nota é um atributo associado ao relacionamento, cada avaliação estará associada a diversos alunos e cada aluno estará associado a diversas avaliações;
- *Disciplina* **Tem** *Avaliação*: registra as disciplinas a qual cada avaliação está associada, uma disciplina pode possuir diversas avaliações e cada avaliação está associada a uma disciplina apenas;
- *Bimestre* **Possui** *Avaliação*: registra a qual bimestre cada avaliação está associada, cada avaliação só pode pertencer a um bimestre e cada bimestre pode possuir diversas avaliações;
- *Aluno* **Associado** *Bimestre*: registra a participação de cada aluno em cada bimestre, cada aluno estará ligado a diversos bimestres e cada bimestre possuirão diversos alunos;
- *Turma* **Realiza** *Avaliação*: registra as avaliações prestadas por cada turma, cada turma pode realizar diversas avaliações e cada avaliação só é realizada por apenas uma turma.

Etapa 2 - Mapeamento lógico e implementação

2.1 - Mapeamento lógico

A seguir estão descritas as tabelas a serem implementadas no banco de dados, em conformidade com o DER anteriormente descrito. Se encontram sublinhadas as chaves primárias e em itálico as chaves estrangeiras.

- Professores(Matricula, CPF, Nome, dtAdmissao, Salario, Email, AreaEnsino, Titulacao, Telefone1, Telefone2);
- EnderecoProfessor(*MatrículaProfessor*, Municipio, Bairro, Numero, Logradouro);
- Secretarios(Matricula, CPF, Nome, dtAdmissao, Salario, Email, TipoCargo, HorarioTrabalho, Telefone1, Telefone2);
- EnderecoSecretario(*MatrículaSecretario*, Municipio, Bairro, Numero, Logradouro);
- Disciplinas(idDisciplina, TotalAulas, CargaHoraria, Nome);
- Turmas(idTurma, Ano, QuantAlunos, Sala, Serie);
- Aulas(idAula, Conteudo, Data, *idTurma*, *idDisciplina*);
- Alunos(Matricula, CPF, Sexo, Nome, DataNascimento, NomeResponsavel, TelefoneResponsavel1, TelefoneResponsavel2, *MatriculaSecretario*);
- EnderecoAlunos(*MatriculaAluno*, Zona, Municipio, Bairro, Numero, Logradouro);
- Avaliacoes(idAvaliacao, Tipo, Data, *idBimestre*, *idDisciplina*, *idTurma*);
- Bimestres(idBimestre, AnoLetivo, Numero, dtInicio, dtTermino);
- Faltas(idFalta, *MatriculaALuno*, Justificativa, Data);
- FaltasPorAula(*idFalta*, *idAula*);
- AvaliacaoDeAluno(*MatriculaAluno*, *idAvaliacao*, Nota);
- BimestreDeAluno(*MatriculaAluno*, *idBimestre*, Situacao);
- TurmasDeAluno(*idTurma*, *MatriculaAluno*, AnoMatriculado);
- ProfessoresPorTurma(*MatriculaProfessor*, *idTurma*, *idDisciplina*);

2.2 - Dicionário lógico de dados

De acordo com as tabelas previamente apresentadas, a seguir se encontra o dicionário lógico para cada tabela e coluna do banco de dados.

Professores: Relação que guarda as informações dos professores				
Atributo	Descrição	Tipo	Domínio	Restrição
Matricula	Armazena a matrícula de cada professor	CHARACTER (8)	Caracteres numéricos	Chave primária
CPF	Armazena o CPF de cada professor, incluindo pontos e hífen	CHARACTER (14)	Caracteres numéricos, ponto e hífen	Atributo chave (único), não nulo
Nome	Armazena o nome de cada professor	VARCHAR (100)	VARCHAR (100)	Não nulo
dtAdmissao	Armazena a data de admissão do professor na instituição	DATE	DATE	Não nulo
Salario	Armazena o salário atual do professor	REAL	Valores positivos	Não nulo
Email	Armazena o email do professor	VARCHAR (30)	VARCHAR (30)	Não nulo
AreaEnsino	Armazena a área de ensino em que o professor atua	VARCHAR (40)	VARCHAR (40)	Não nulo
Titulacao	Armazena a titulação acadêmica do professor	VARCHAR (40)	VARCHAR (40)	Não nulo
Telefone1	Armazena o primeiro telefone do professor	VARCHAR (15)	Caracteres que representam números	Não nulo
Telefone2	Armazena o segundo telefone do professor	VARCHAR (15)	Caracteres que representam números	Nenhuma

EnderecoProfessor: Relação que guarda os endereços de cada professor				
Atributo	Descrição	Tipo	Domínio	Restrição
MatriculaProfessor	Armazena a matrícula do professor referente ao endereço	CHARACTER (8)	Caracteres numéricos	Chave primária, Chave estrangeira referenciando tabela Professor
Municipio	Armazena o município onde mora o professor	VARCHAR (30)	VARCHAR (30)	Não nulo
Bairro	Armazena o bairro onde mora o professor	VARCHAR (30)	VARCHAR (30)	Não nulo
Numero	Armazena o número da residência onde mora o professor	INT	Números inteiros positivos	Não nulo
Logradouro	Armazena o logradouro onde mora o professor	VARCHAR (30)	VARCHAR (30)	Não nulo

Secretarios: Relação que armazena as informações dos secretários				
Atributo	Descrição	Tipo	Domínio	Restrição
Matricula	Armazena a matrícula de cada secretário	CHARACTER (8)	Caracteres numéricos	Chave primária
CPF	Armazena o CPF de cada secretário	CHARACTER (14)	Caracteres numéricos, ponto e hífen	Atributo chave (único), não nulo
Nome	Armazena o nome de cada secretário	VARCHAR (100)	VARCHAR (100)	Não nulo
dtAdmissao	Armazena a data de admissão do secretário na instituição	DATE	DATE	Não nulo

Salario	Armazena o salário atual do secretário	REAL	Valores positivos	Não nulo
Email	Armazena o email do secretário	VARCHAR (30)	VARCHAR (30)	Não nulo
TipoCargo	Armazena a atribuição de cargo específica do secretário	VARCHAR (30)	VARCHAR (30)	Não nulo
HorarioTrabalho	Armazena o horário do expediente de trabalho do secretário	VARCHAR (30)	VARCHAR (30)	Não nulo
Telefone1	Armazena o primeiro telefone do secretário	VARCHAR (15)	Caracteres que representam números	Não nulo
Telefone2	Armazena o segundo telefone do secretário	VARCHAR (15)	Caracteres que representam números	Nenhuma

EnderecoSecretario: Relação que armazena os endereços dos secretários				
Atributo	Descrição	Tipo	Domínio	Restrição
MatriculaSecretário	Armazena a matrícula do secretário referente ao endereço	CHARACTER (8)	Caracteres numéricos	Chave primária, Chave estrangeira referenciando tabela Secretário
Municipio	Armazena o município onde mora o secretário	VARCHAR (30)	VARCHAR (30)	Não nulo
Bairro	Armazena o bairro onde mora o secretário	VARCHAR (30)	VARCHAR (30)	Não nulo

Numero	Armazena o número da residência onde mora o secretário	INT	Números inteiros positivos	Não nulo
Logradouro	Armazena o logradouro onde mora o secretário	VARCHAR (30)	VARCHAR (30)	Não nulo

Disciplinas: Relação que armazena as informações das disciplinas ofertadas				
Atributo	Descrição	Tipo	Domínio	Restrição
idDisciplina	Armazena o identificador da disciplina	INT	Inteiros positivos	Chave primária
TotalAulas	Armazena o total de aulas semanais de uma disciplina	INT	Inteiros positivos	Não nulo
CargaHoraria	Armazena a carga horária anual de uma disciplina	INT	Inteiros positivos	Não nulo
Nome	Armazena o nome da disciplina	VARCHAR(20)	VARCHAR(20)	Não nulo

Turmas: Relação que armazena as informações das turmas cadastradas				
Atributo	Descrição	Tipo	Domínio	Restrição
idTurma	Armazena o identificador de cada turma	INT	Inteiros positivos	Chave primária
Ano	Armazena o ano em que essa turma foi formada	INT	Inteiros positivos que representem anos	Não nulo
QuantAlunos	Armazena a quantidade de alunos matriculados na turma	INT	Inteiros positivos	Não nulo

Sala	Armazena a localização da sala onde a turma se localiza	VARCHAR (40)	VARCHAR (40)	Não nulo
Serie	Armazena a série da turma	VARCHAR (20)	VARCHAR (20)	Não nulo

Aulas: Relação que armazena as informações das aulas registradas				
Atributo	Descrição	Tipo	Domínio	Restrição
idAula	Armazena o identificador da aula	INT	Inteiros positivos	Chave primária
Conteudo	Armazena conteúdo ministrado na aula	VARCHAR (100)	VARCHAR (100)	Não nulo
Data	Armazena a data da aula	DATE	DATE	Não nulo
idTurma	Armazena a turma que participou da aula	INT	Inteiros positivos	Chave estrangeira referenciando a tabela Turmas
idDisciplina	Armazena a disciplina a qual pertence a aula	INT	Inteiros positivos	Chave estrangeira referenciando a tabela Disciplinas

Alunos: Relação que armazena as informações sobre os alunos matriculados				
Atributo	Descrição	Tipo	Domínio	Restrição
Matricula	Armazena a matrícula de cada aluno	CHARACTER (8)	Caracteres numéricos	Chave primária
CPF	Armazena o CPF de cada secretário	CHARACTER (14)	Caracteres numéricos, ponto e hífen	Atributo chave (único), não nulo
Sexo	Armazena o sexo de cada aluno	CHARACTER (1)	CHARACTER (1)	Não nulo

Nome	Armazena o nome de cada secretário	VARCHAR (100)	VARCHAR (100)	Não nulo
DataNascimento	Armazena a data de nascimento de cada aluno	DATE	DATE	Não nulo
NomeResponsável	Armazena o nome do responsável de cada aluno	VARCHAR (100)	VARCHAR (100)	Não nulo
TelefoneResponsavel1	Armazena o primeiro telefone do responsável	VARCHAR (15)	Caracteres que representam números	Não nulo
TelefoneResponsavel2	Armazena o segundo telefone do responsável	VARCHAR (15)	Caracteres que representam números	Nenhuma
MatriculaSecretario	Armazena a matrícula do secretário que matriculou o aluno	CHARACTER (8)	Caracteres numéricos	Chave estrangeira

EnderecoAlunos: Relação que armazena os endereços dos alunos				
Atributo	Descrição	Tipo	Domínio	Restrição
MatriculaAluno	Armazena a matrícula do aluno referente ao endereço	CHARACTER (8)	Caracteres numéricos	Chave primária, Chave estrangeira referenciando tabela Alunos
Zona	Armazena a zona do município onde mora o aluno	CHARACTER (20)	CHARACTER (20)	Não nulo
Municipio	Armazena o município onde mora o aluno	VARCHAR (30)	VARCHAR (30)	Não nulo
Bairro	Armazena o bairro onde mora o aluno	VARCHAR (30)	VARCHAR (30)	Não nulo

Numero	Armazena o número da residência onde mora o aluno	INT	Números inteiros positivos	Não nulo
Logradouro	Armazena o logradouro onde mora o aluno	VARCHAR (30)	VARCHAR (30)	Não nulo

Bimestres: Relação que armazena as informações de cada bimestre letivo				
Atributo	Descrição	Tipo	Domínio	Restrição
idBimestre	Armazena o identificador de cada bimestre	INT	Inteiros positivos	Chave primária
AnoLetivo	Armazena o ano letivo do bimestre	INT	Inteiros positivos	Não nulo
Numero	Armazena o número do bimestre no ano	INT	Inteiros positivos	Não Nulo
dtInicio	Armazena a data de início do bimestre	DATE	DATE	Nenhuma
dtTermino	Armazena a data de término do bimestre	DATE	DATE	Nenhuma

Avaliacoes: Relação que armazena as informações das avaliações realizadas				
Atributo	Descrição	Tipo	Domínio	Restrição
idAvaliacao	Armazena o identificador de cada avaliação	INT	Inteiros positivos	Chave primária
Tipo	Armazena o tipo da avaliação	VARCHAR (30)	VARCHAR (30)	Não nulo
Data	Armazena a data	DATE	DATE	Não nulo
idBimestre	Armazena o identificador do bimestre referente a avaliação	INT	Inteiros positivos	Chave estrangeira referenciando a tabela Bimestres

idDisciplina	Armazena o identificador da disciplina referente à avaliação	INT	Inteiros positivos	Chave estrangeira referenciando a tabela Disciplinas
idTurma	Armazena o identificador turma para qual é realizada a avaliação	INT	Inteiros positivos	Chave estrangeira referenciando a tabela Turmas

Faltas: Relação que armazena as informações das faltas de cada aluno				
Atributo	Descrição	Tipo	Domínio	Restrição
idFalta	Identificador de cada falta	INT	Inteiros positivos	Chave primária
MatriculaAluno	Armazena a matrícula do aluno referente a falta	CHARACTER (8)	Caracteres numéricos	Chave primária, chave estrangeira referenciando a tabela Alunos
Justificativa	Armazena a justificativa da falta	VARCHAR (100)	VARCHAR (100)	Nenhuma
Data	Armazena a data da falta	DATE	DATE	Não nulo

FaltasPorAula: Relação que armazena as aulas que foram perdidas para cada falta registrada				
Atributo	Descrição	Tipo	Domínio	Restrição
idFalta	Armazena o identificador da falta associada a aula	INT	Inteiros positivos	Chave primária, chave estrangeira
idAula	Armazena o identificador da aula associada a falta	INT	Inteiros positivos	Chave primária, chave estrangeira

AvaliacaoDeAluno: Relação que armazena as informações dos resultados de cada avaliação prestada por cada aluno				
Atributo	Descrição	Tipo	Domínio	Restrição
MatriculaAluno	Armazena a matrícula do aluno referente a falta	CHARACTER (8)	Caracteres numéricos	Chave primária, chave estrangeira referenciando a tabela Alunos
idAvaliacao	Armazena o identificador referente a avaliação	INT	Inteiros positivos	Chave primária, chave estrangeira referenciando a tabela Avaliacoes
Nota	Armazena a nota da avaliação	REAL	Valores positivos	Não nulo

BimestreDeAluno: Relação que armazena os bimestres letivos que cada aluno cursou				
Atributo	Descrição	Tipo	Domínio	Restrição
MatriculaAluno	Armazena a matrícula do aluno referente ao bimestre	CHARACTER (8)	Caracteres numéricos	Chave primária, chave estrangeira referenciando a tabela Alunos
idBimestre	Armazena o identificador do bimestre	INT	Inteiros positivos	Chave primária, chave estrangeira referenciando a tabela Bimestres
Situacao	Armazena a situação de cada aluno no devido bimestre	VARCHAR (20)	VARCHAR (20)	Nenhuma

TurmasDeAluno: Relação que armazena em que turma cada aluno está matriculado				
Atributo	Descrição	Tipo	Domínio	Restrição
idTurma	Armazena a turma que o aluno pertence	INT	Inteiros positivos	Chave primária, chave estrangeira referenciando a tabela Turmas

MatriculaAluno	Armazena a matrícula do aluno referente à turma	CHARACTER (8)	Caracteres numéricos	Chave primária, chave estrangeira referenciando a tabela Alunos
AnoMatriculado	Armazena o ano em que o aluno foi matriculado naquela turma	INT	Inteiros positivos que representem anos	Não nulo

ProfessoresPorTurma: Relação que armazena as informações sobre as disciplinas ministradas por professores para cada turma				
Atributo	Descrição	Tipo	Domínio	Restrição
MatriculaProfessor	Armazena a matrícula do professor que ministra a disciplina	CHARACTER (8)	Caracteres numéricos	Chave primária, Chave estrangeira referenciando tabela Professor
idTurma	Armazena o identificador turma para qual é ministrada a disciplina	INT	Inteiros positivos	Chave estrangeira referenciando a tabela Turmas
idDisciplina	Armazena o identificador da disciplina ministrada	INT	Inteiros positivos	Chave estrangeira referenciando a tabela Disciplinas

2.3 - Scripts de criação das tabelas

A seguir o script de criação das tabelas no banco de dados. Como no PostgreSQL o padrão é que não haja case-sensitive, os nomes de tabelas e colunas serão

Criação da tabela “professores”:

```
CREATE TABLE professores (
    matricula CHARACTER(8),
    cpf CHARACTER(14) UNIQUE,
    nome VARCHAR(100) NOT NULL,
    dtadmissao DATE NOT NULL,
    salario REAL NOT NULL,
    email VARCHAR(30) NOT NULL,
    areaensino VARCHAR(40) NOT NULL,
```

```
    titulacao VARCHAR(40) NOT NULL,
    telefone1 VARCHAR(15) NOT NULL,
    telefone2 VARCHAR(15),
    PRIMARY KEY (matricula)
);
```

Criação da tabela “enderecoProfessor”:

```
CREATE TABLE enderecoprofessor (
    matriculaprofessor CHARACTER(8),
    municipio VARCHAR(30) NOT NULL,
    bairro VARCHAR(30) NOT NULL,
    numero INTEGER NOT NULL,
    logradouro VARCHAR(30) NOT NULL,
    PRIMARY KEY (matriculaprofessor),
    FOREIGN KEY (matriculaprofessor) REFERENCES professores(matricula)
);
```

Criação da tabela “secretarios”;

```
CREATE TABLE secretarios (
    matricula CHARACTER(8),
    cpf CHARACTER(14) UNIQUE,
    nome VARCHAR(100) NOT NULL,
    dtadmissao DATE NOT NULL,
    salario REAL NOT NULL,
    email VARCHAR(30) NOT NULL,
    tipocargo VARCHAR(30) NOT NULL,
    horariotrabalho VARCHAR(30) NOT NULL,
    telefone1 VARCHAR(15) NOT NULL,
    telefone2 VARCHAR(15),
    PRIMARY KEY (matricula)
);
```

Criação da tabela “enderecosecretario”:

```
CREATE TABLE enderecosecretario (
    matriculasecretario CHARACTER(8),
    municipio VARCHAR(30) NOT NULL,
    bairro VARCHAR(30) NOT NULL,
    numero INTEGER NOT NULL,
    logradouro VARCHAR(30) NOT NULL,
    primary key (matriculasecretario),
    foreign key (matriculasecretario) REFERENCES secretarios(matricula)
);
```

Criação da tabela “disciplinas”:

```
CREATE TABLE disciplinas (
    iddisciplina INTEGER,
    totalaulas INTEGER NOT NULL,
    cargahoraria INTEGER NOT NULL,
    nome VARCHAR(20) NOT NULL,
```

```
    PRIMARY KEY (iddisciplina)
);
```

Criação da tabela “turmas”:

```
CREATE TABLE turmas (
    idturma INTEGER,
    ano INTEGER NOT NULL,
    quantalunos INTEGER NOT NULL,
    sala VARCHAR(40) NOT NULL,
    serie VARCHAR(20) NOT NULL,
    PRIMARY KEY (idturma)
);
```

Criação da tabela “aulas”:

```
CREATE TABLE aulas (
    idaula INTEGER,
    conteudo VARCHAR(100) NOT NULL,
    data DATE NOT NULL,
    idturma INTEGER,
    iddisciplina INTEGER,
    PRIMARY KEY (idaula),
    FOREIGN KEY (idturma) REFERENCES turmas(idturma),
    FOREIGN KEY (iddisciplina) REFERENCES disciplinas(iddisciplina)
);
```

Criação da tabela “alunos”:

```
CREATE TABLE alunos (
    matricula CHARACTER(8),
    cpf CHARACTER(14) UNIQUE,
    sexo CHARACTER(1) NOT NULL,
    nome VARCHAR(100) NOT NULL,
    datanascimento DATE NOT NULL,
    nomeresponsavel VARCHAR(100) NOT NULL,
    telefoneresponsavel1 VARCHAR(15) NOT NULL,
    telefoneresponsavel2 VARCHAR(15),
    matriculasecretario CHARACTER(8),
    PRIMARY KEY (matricula),
    FOREIGN KEY (matriculasecretario) REFERENCES secretarios(matricula)
);
```

Criação da tabela “enderecoalunos”:

```
CREATE TABLE enderecoalunos(
    matriculaaluno CHARACTER(8),
    zona CHARACTER(20),
    municipio VARCHAR(30) NOT NULL,
    bairro VARCHAR(30) NOT NULL,
    numero INTEGER NOT NULL,
    logradouro VARCHAR(30) NOT NULL,
    PRIMARY KEY (matriculaaluno),
```

```
    FOREIGN KEY (matriculaaluno) REFERENCES alunos(matricula)
);
```

Criação da tabela “bimestres”:

```
CREATE TABLE bimestres (
    idbimestre INTEGER,
    anoletivo INTEGER NOT NULL,
    numero INTEGER NOT NULL,
    dtinicio DATE,
    dttermino DATE,
    PRIMARY KEY (idbimestre)
);
```

Criação da tabela “avaliacoes”:

```
CREATE TABLE avaliacoes (
    idavaliacao INTEGER,
    tipo varchar(30) NOT NULL,
    data date NOT NULL,
    idbimestre INTEGER,
    iddisciplina INTEGER,
    idturma INTEGER,
    PRIMARY KEY (idavaliacao),
    FOREIGN KEY (idbimestre) REFERENCES bimestres(idbimestre),
    FOREIGN KEY (iddisciplina) REFERENCES disciplinas(iddisciplina),
    FOREIGN KEY (idturma) REFERENCES turmas(idturma)
);
```

Criação da tabela “faltas”:

```
CREATE TABLE faltas (
    idfalta INTEGER,
    matriculaaluno CHARACTER(8),
    justificativa VARCHAR(100) NOT NULL,
    data DATE NOT NULL,
    PRIMARY KEY (idfalta),
    FOREIGN KEY (matriculaaluno) REFERENCES alunos(matricula)
);
```

Criação da tabela “faltasporaula”:

```
CREATE TABLE faltasporaula (
    idfalta INTEGER,
    idaula INTEGER,
    PRIMARY KEY (idfalta, idaula),
    FOREIGN KEY (idfalta) REFERENCES faltas(idfalta),
    FOREIGN KEY (idaula) REFERENCES aulas(idaula)
);
```

Criação da tabela “avaliacaodealuno”:

```
CREATE TABLE avaliacaodealuno (
```

```
    matriculaaluno CHARACTER(8),
    idavaliacao INTEGER,
    nota REAL NOT NULL,
    PRIMARY KEY (matriculaaluno, idavaliacao),
    FOREIGN KEY (matriculaaluno) REFERENCES alunos(matricula),
    FOREIGN KEY (idavaliacao) REFERENCES avaliacoes(idavaliacao)
);
```

Criação da tabela “bimestredealuno”:

```
CREATE TABLE bimestredealuno (
    matriculaaluno CHARACTER(8),
    idbimestre INTEGER,
    situacao VARCHAR(20),
    PRIMARY KEY (matriculaaluno, idbimestre),
    FOREIGN KEY (matriculaaluno) REFERENCES alunos(matricula),
    FOREIGN KEY (idbimestre) REFERENCES bimestres(idbimestre)
);
```

Criação da tabela “turmasdealuno”:

```
CREATE TABLE turmasdealuno (
    idturma INTEGER,
    matriculaaluno CHARACTER(8),
    anomatriculado INTEGER NOT NULL,
    PRIMARY KEY (idturma, matriculaaluno),
    FOREIGN KEY (idturma) REFERENCES turmas(idturma),
    FOREIGN KEY (matriculaaluno) REFERENCES alunos(matricula)
);
```

Criação da tabela “professoresporturma”:

```
CREATE TABLE professoresporturma (
    matriculaprofessor CHARACTER(8),
    idturma INTEGER,
    iddisciplina INTEGER,
    PRIMARY KEY (matriculaprofessor, idturma, iddisciplina),
    FOREIGN KEY (matriculaprofessor) REFERENCES professores(matricula),
    FOREIGN KEY (idturma) REFERENCES turmas(idturma),
    FOREIGN KEY (iddisciplina) REFERENCES disciplinas(iddisciplina)
);
```

2.4 - Scripts de povoamento das tabelas

Povoamento da tabela “professores”:

```
INSERT INTO professores
VALUES
('20210001', '111.111.111-11', 'José Antônio da Silva', '2021-02-03', 2000.00,
'joseantonio@gmail.com', 'Letras em Português', 'Graduação', '83982074951',
'83981080123'),
```

```
('20210002', '222.222.222-22', 'Cândida Maria Mariz', '2021-02-03', 4500.00,
'candidamaria@gmail.com', 'Matemática e suas Técnologias', 'Mestrado', '83988764532',
'83988826354'),
('20200003', '333.333.333-33', 'José Guimarães da Silva Neto', '2020-02-03', 2500.00,
'joseguimaraes@gmail.com', 'Ciências Humanas e Sociais', 'Graduação', '83991234572',
'83991876512');

INSERT INTO professores
VALUES
('20180004', '444.444.444-44', 'Luciamara Cavalcante Araújo', '2018-02-03', 2700.00,
'luciamaracavalcante@gmail.com', 'Ciências da Natureza', 'Graduação', '83981768900'),
('20190005', '555.555.555-55', 'Paulo Freitas da Silva', '2019-02-03', 1700.00,
'paulofreitas@gmail.com', 'História Geral', 'Graduação', '83982072233');
```

Povoamento da tabela “enderecoprofessor”:

```
INSERT INTO enderecoprofessor
VALUES
('20210001', 'Sousa', 'Estação', 401, 'Rua Dom Pedro I'),
('20210002', 'Sousa', 'Estação', 304, 'Rua Dom Pedro II'),
('20200003', 'Sousa', 'Centro', 202, 'Rua Gualberto Filho'),
('20180004', 'Cajazeiras', 'Centro', 150, 'Avenida Joca Claudino'),
('20190005', 'Cajazeiras', 'Por do Sol', 989, 'Rua José Leite de Oliveira');
```

Povoamento da tabela “secretarios”:

```
INSERT INTO secretarios
VALUES
('20150021', '666.666.666-66', 'Ranyel Soares', '2015-05-09', 2500.00,
'ranyelsoares@gmail.com', 'Comissionado', 'Integral', '83981234567', '83981876345'),
('20160023', '777.777.777-77', 'Matheus Lorenzo Braga', '2016-03-07', 3000.00,
'matheuslorenzo@gmail.com', 'Comissionado', 'Matutino', '83988342515', '83981231487'),
('19980002', '888.888.888-88', 'Maria Rita Sousa', '1998-01-02', 2000.00,
'mariarita@gmail.com', 'Concursado', 'Integral', '83991456789','83991887734'),
('20200008', '999.999.999-99', 'Antônia Carla Andrade', '2020-04-04', 3000.00,
'antoniacarla@gmail.com', 'Comissionado', 'Vespertino', '83982564788', '83981236789'),
('20210003', '123.123.123-12', 'Carlos Alberto Nobrega', '2021-01-02', 3500.00,
'carlosalberto@gmail.com', 'Concursado', 'Noturno', '83999876234', '83982557890');
```

Povoamento da tabela “enderecosecretario”:

```
INSERT INTO enderecosecretario
VALUES
('20150021', 'Marizópolis', 'Santo Antônio', 98, 'Rua 7 de Setembro'),
('20160023', 'Sousa', 'Centro', 203, 'Rua Gualberto Filho'),
('19980002', 'Marizópolis', 'Centro', 54, 'Luzia Braga'),
('20200008', 'Cajazeiras', 'Centro', 149, 'Avenida Joca Claudino'),
('20210003', 'Marizópolis', 'Santo Antônio', 23, 'Rua 7 de Setembro');
```

Povoamento da tabela “disciplinas”:

```
INSERT INTO disciplinas
VALUES
```

```
(1, 8, 300, 'Matemática'),
(2, 8, 300, 'Português'),
(3, 5, 200, 'Ciências'),
(4, 4, 170, 'História'),
(5, 4, 170, 'Geografia');
```

Povoamento da tabela “turmas”:

```
INSERT INTO turmas
VALUES
(1, 2023, 30, 'Sala 101', '6º ano A'),
(2, 2023, 28, 'Sala 102', '6º ano B'),
(3, 2023, 32, 'Sala 103', '7º ano'),
(4, 2023, 30, 'Sala 201', '8 ano'),
(5, 2023, 28, 'Sala 202', '9º ano A'),
(6, 2023, 32, 'Sala 203', '9º ano B');
```

Povoamento da tabela “aulas”:

```
INSERT INTO aulas
VALUES
(1, 'Introdução à Matemática', '2023-01-05', 1, 1),
(2, 'Gramática e Redação', '2023-01-06', 2, 2),
(3, 'Ciências Naturais', '2023-01-07', 3, 3),
(4, 'História do Brasil', '2023-01-08', 4, 4),
(5, 'Geografia Mundial', '2023-01-09', 5, 5),
(6, 'Álgebra Avançada', '2023-01-10', 1, 1),
(7, 'Leitura e Escrita', '2023-01-11', 2, 2),
(8, 'Física e Química', '2023-01-12', 3, 3),
(9, 'Civilizações Antigas', '2023-01-13', 4, 4),
(10, 'Clima e Meio Ambiente', '2023-01-14', 5, 5),
(11, 'Geometria Básica', '2023-01-15', 1, 1),
(12, 'Leitura e Interpretação de Textos', '2023-01-16', 2, 2),
(13, 'Biologia Celular', '2023-01-17', 3, 3),
(14, 'História Antiga', '2023-01-18', 4, 4),
(15, 'Problemas Ambientais Globais', '2023-01-19', 5, 5),
(16, 'Estatísticas e Probabilidades', '2023-01-20', 1, 1),
(17, 'Redação Criativa', '2023-01-21', 2, 2),
(18, 'Química Orgânica', '2023-01-22', 3, 3),
(19, 'Idade Média', '2023-01-23', 4, 4),
(20, 'Ecologia e Sustentabilidade', '2023-01-24', 5, 5);
```

Povoamento da tabela “alunos”:

```
INSERT INTO alunos
VALUES
('20230021', '121.121.121-12', 'F', 'Maria Clara', '2010-03-15', 'José Alceu', '83981008786',
'83981234588', '20150021'),
('20230022', '232.232.232-23', 'F', 'Ana Luzia', '2009-07-20', 'Marta Araújo', '83982768594',
'83982345697', '20150021'),
('20230024', '343.343.343-34', 'M', 'José Francisco', '2008-01-10', 'Lurdes Sousa',
'84982348799', '83982008765', '20160023'),
```

```
('20230025', '454.454.454-45', 'F', 'Ana Lucia', '2007-11-05', 'José Alceu', '83981008786',
'83981234588', '20150021'),
('20230026', '565.565.565-56', 'M', 'Matheus Francisco', '2006-05-18', 'Marta Araújo',
'83982768594', '83982345697', '20150021'),
('20230027', '676.676.676-67', 'F', 'Viviane Sarmento', '2005-09-30', 'Marta Araújo',
'83982768594', '83982345697', '20150021'),
('20230028', '787.787.787-78', 'M', 'José Sicupira Neto', '2004-12-22', 'Edmar Filho',
'83981454536', '83982075657','20200008'),
('20230029', '889.889.889-89', 'F', 'Vitória Braga', '2003-08-17', 'Marta Araújo',
'83982768594', '83982345697', '20150021'),
('20230030', '990.990.990-90', 'M', 'João Vítor', '2002-06-25',  'Edmar Filho',
'83981454536', '83982075657','20200008'),
('20230031', '103.103.103-02', 'F', 'Adrielly Sousa', '2001-04-09', 'Hernandes Lopes',
'83981234572', '83988346277', '20210003'),
('20230032', '234.234.234-23', 'M', 'Luiz Felipe', '2000-10-12', 'Ozenir Sousa',
'83982073981', '83982008765', '20150021'),
('20230033', '345.345.345-34', 'M', 'Karlos Messyas', '1999-03-29', 'Ozenir Sousa',
'83982073981', '83982008765', '20150021'),
('20230034', '456.406.456-45', 'M', 'Flávio Túlio', '2014-07-03',  'Edmar Filho',
'83981454536', '83982075657','20200008'),
('20230035', '567.567.567-56', 'M', 'Guilherme Abreu', '2013-05-12',  'Ozenir Sousa',
'83982073981', '83982008765', '20150021'),
('20230036', '678.678.678-67', 'M', 'Matheus Jorge Filho', '2012-08-22', 'Martha Sousa e
Silva', '83988767876', '83988765422', '20200008'),
('20230037', '789.789.789-78', 'F', 'Alice Estrela', '2011-10-10', 'Martha Sousa e Silva',
'83988767876', '83988765422', '20200008'),
('20230038', '890.890.890-89', 'M', 'Esdras Gabriel', '2010-02-14',  'Edmar Filho',
'83981454536', '83982075657','20200008'),
('20230039', '901.901.901-90', 'M', 'Marcos Paulo', '2009-04-19',  'Edmar Filho',
'83981454536', '83982075657','20200008'),
('20230040', '012.012.012-01', 'M', 'André Marcos', '2008-11-26', 'Marta Araújo',
'83982768594', '83982345697', '20150021'),
('20230041', '123.123.123-45', 'F', 'Analucia Estrela Silva', '2007-03-30', 'José Alceu',
'83981008786', '83981234588', '20150021'),
('20230042', '234.234.234-56', 'M', 'José Anuário', '2006-09-02', 'Carla Anuário',
'83988423456', '83981080706', '19980002'),
('20230043', '345.345.345-67', 'F', 'Maria Anuário', '2005-01-05', 'Carla Anuário',
'83988423456', '83981080706', '19980002'),
('20230044', '456.456.456-78', 'M', 'Alice Esther', '2004-04-10', 'José Alceu',
'83981008786', '83981234588', '20150021'),
('20230045', '567.567.567-89', 'F', 'Matheus Gabriel Bamboa', '2003-06-15', 'José Alceu',
'83981008786', '83981234588', '20150021');
```

Povoamento da tabela “enderecoalunos”:

```
INSERT INTO enderecoalunos
VALUES
('20230021', 'Urbana', 'Sousa', 'Gato Preto', 11, 'Rua Josival Messyas'),
('20230025', 'Urbana', 'Sousa', 'Gato Preto', 11, 'Rua Josival Messyas'),
('20230041', 'Urbana', 'Sousa', 'Gato Preto', 11, 'Rua Josival Messyas'),
('20230044', 'Urbana', 'Sousa', 'Gato Preto', 11, 'Rua Josival Messyas'),
('20230045', 'Urbana', 'Sousa', 'Gato Preto', 11, 'Rua Josival Messyas'),
('20230022', 'Rural', 'Sousa', 'Sítio dos Estrelas', 10, 'Rua Vereador Gabriel Marques'),
('20230026', 'Rural', 'Sousa', 'Sítio dos Estrelas', 10, 'Rua Vereador Gabriel Marques'),
```

```
('20230029', 'Rural', 'Sousa', 'Sítio dos Estrelas', 10, 'Rua Vereador Gabriel Marques'),
('20230040', 'Rural', 'Sousa', 'Sítio dos Estrelas', 10, 'Rua Vereador Gabriel Marques'),
('20230027', 'Rural', 'Sousa', 'Sítio dos Estrelas', 10, 'Rua Vereador Gabriel Marques'),
('20230024', 'Urbana', 'Sousa', 'Gato Preto', 12, 'Rua Josival Messyas'),
('20230028', 'Rural', 'São Francisco', 'Duas lagoas', 9, 'Rua Nova Vida'),
('20230030', 'Rural', 'São Francisco', 'Duas lagoas', 9, 'Rua Nova Vida'),
('20230034', 'Rural', 'São Francisco', 'Duas lagoas', 9, 'Rua Nova Vida'),
('20230038', 'Rural', 'São Francisco', 'Duas lagoas', 9, 'Rua Nova Vida'),
('20230039', 'Rural', 'São Francisco', 'Duas lagoas', 9, 'Rua Nova Vida'),
('20230031', 'Urbana', 'Sousa', 'Centro', 12, 'Rua Nova'),
('20230032', 'Urbana', 'Marizópolis', 'Centro', 45, 'Rua das Alamedas'),
('20230033', 'Urbana', 'Marizópolis', 'Centro', 45, 'Rua das Alamedas'),
('20230035', 'Urbana', 'Marizópolis', 'Centro', 45, 'Rua das Alamedas'),
('20230036', 'Rural', 'Sousa', 'Sítio dos Dinossauros', 23, 'Rua Asfaltada'),
('20230037', 'Rural', 'Sousa', 'Sítio dos Dinossauros', 23, 'Rua Asfaltada'),
('20230042', 'Urbana', 'Sousa', 'Gato Preto', 22, 'Rua Josival Messyas'),
('20230043', 'Urbana', 'Sousa', 'Gato Preto', 22, 'Rua Josival Messyas');
```

Povoamento da tabela “bimestres”:

```
INSERT INTO bimestres
VALUES
(20231, 2023, 1, '2023-02-01', '2023-03-31'),
(20232, 2023, 2, '2023-04-03', '2023-05-31'),
(20233, 2023, 3, '2023-06-01', '2023-07-31'),
(20234, 2023, 4, '2023-08-01', '2023-09-29'),
(20235, 2023, 5, '2023-10-02', '2023-11-30');
```

Povoamento da tabela “avaliacoes”:

```
INSERT INTO avaliacoes
VALUES
(1, 'Prova', '2023-03-15', 20231, 2, 1),
(2, 'Prova', '2023-03-15', 20231, 2, 2),
(3, 'Projeto', '2023-03-15', 20231, 2, 3),
(4, 'Projeto', '2023-03-15', 20231, 2, 4),
(5, 'Prova', '2023-03-15', 20231, 2, 5),
(6, 'Prova', '2023-03-15', 20231, 2, 6),
(7, 'Prova', '2023-03-15', 20231, 1, 1),
(8, 'Prova', '2023-03-15', 20231, 1, 2),
(9, 'Fichário', '2023-03-15', 20231, 1, 3),
(10, 'Fichário', '2023-03-15', 20231, 1, 4),
(11, 'Prova', '2023-03-15', 20231, 1, 5),
(12, 'Prova', '2023-03-15', 20231, 1, 6),
(13, 'Prova', '2023-03-15', 20231, 3, 1),
(14, 'Prova', '2023-03-15', 20231, 3, 2),
(15, 'Prova', '2023-03-15', 20231, 3, 3),
(16, 'Projeto', '2023-03-15', 20231, 3, 4),
(17, 'Projeto', '2023-03-15', 20231, 3, 5),
(18, 'Projeto', '2023-03-15', 20231, 3, 6),
(19, 'Prova', '2023-03-15', 20231, 5, 1),
(20, 'Prova', '2023-03-15', 20231, 5, 2),
(21, 'Prova', '2023-03-15', 20231, 5, 3),
```

```
(22, 'Prova', '2023-03-15', 20231, 5, 4),
(23, 'Projeto', '2023-03-15', 20231, 5, 5),
(24, 'Projeto', '2023-03-15', 20231, 5, 6),
(25, 'Projeto', '2023-03-15', 20231, 4, 1),
(26, 'Fichário', '2023-03-15', 20231, 4, 2),
(27, 'Prova', '2023-03-15', 20231, 4, 3),
(28, 'Prova', '2023-03-15', 20231, 4, 4),
(29, 'Projeto', '2023-03-15', 20231, 4, 5),
(30, 'Projeto', '2023-03-15', 20231, 4, 6);
```

Povoamento da tabela “faltas”:

```
INSERT INTO faltas
VALUES
(1, '20230021', 'Doente', '2023-02-15'),
(2, '20230026', 'Doente', '2023-03-17'),
(3, '20230029', 'Doente', '2023-02-10'),
(4, '20230033', 'Perna quebrada', '2023-03-10'),
(5, '20230036', 'Doente', '2023-02-09');
```

Povoamento da tabela “faltasporaula”:

```
INSERT INTO faltasporaula
VALUES
(1, 7),
(2, 4),
(3, 8),
(4, 14),
(5, 12);
```

Povoamento da tabela “turmasdealuno”:

```
INSERT INTO turmasdealuno
VALUES
(1, '20230021', 2023),
(1, '20230022', 2023),
(1, '20230024', 2023),
(1, '20230025', 2023),
(2, '20230026', 2023),
(2, '20230027', 2023),
(2, '20230028', 2023),
(2, '20230029', 2023),
(3, '20230030', 2023),
(3, '20230031', 2023),
(3, '20230032', 2023),
(3, '20230033', 2023),
(4, '20230034', 2023),
(4, '20230035', 2023),
(4, '20230036', 2023),
(4, '20230037', 2023),
(5, '20230038', 2023),
(5, '20230039', 2023),
(5, '20230040', 2023),
```

```
(5, '20230041', 2023),
(6, '20230042', 2023),
(6, '20230043', 2023),
(6, '20230044', 2023),
(6, '20230045', 2023);
```

Povoamento da tabela “avaliacaodealuno”:

```
INSERT INTO avaliacaodealuno
VALUES
('20230021', 1, 8),
('20230021', 7, 9),
('20230021', 13, 10),
('20230021', 19, 6),
('20230021', 25, 4),
('20230022', 1, 3),
('20230022', 7, 7),
('20230022', 13, 8),
('20230022', 19, 7),
('20230022', 25, 10),
('20230024', 1, 8),
('20230024', 7, 8),
('20230024', 13, 9),
('20230024', 19, 10),
('20230024', 25, 10),
('20230025', 1, 4),
('20230025', 7, 3),
('20230025', 13, 9),
('20230025', 19, 9),
('20230025', 25, 10),
('20230026', 2, 7),
('20230026', 8, 7),
('20230026', 14, 8),
('20230026', 20, 9),
('20230026', 26, 2),
('20230027', 2, 4),
('20230027', 8, 6),
('20230027', 14, 8),
('20230027', 20, 2),
('20230027', 26, 10),
('20230028', 2, 10),
('20230028', 8, 10),
('20230028', 14, 9),
('20230028', 20, 8),
('20230028', 26, 9),
('20230029', 2, 7),
('20230029', 8, 6),
('20230029', 14, 6),
('20230029', 20, 10),
('20230029', 26, 10),
('20230030', 3, 10),
('20230030', 9, 10),
('20230030', 15, 10),
('20230030', 21, 9),
```

('20230030', 27, 8),
('20230031', 3, 9),
('20230031', 9, 9),
('20230031', 15, 8),
('20230031', 21, 3),
('20230031', 27, 4),
('20230032', 3, 7),
('20230032', 9, 7),
('20230032', 15, 8),
('20230032', 21, 8),
('20230032', 27, 9),
('20230033', 3, 8),
('20230033', 9, 6),
('20230033', 15, 10),
('20230033', 21, 4),
('20230033', 27, 6),
('20230034', 4, 6),
('20230034', 10, 7),
('20230034', 16, 4),
('20230034', 22, 10),
('20230034', 28, 9),
('20230035', 4, 5),
('20230035', 10, 3),
('20230035', 16, 7),
('20230035', 22, 6),
('20230035', 28, 10),
('20230036', 4, 2),
('20230036', 10, 4),
('20230036', 16, 9),
('20230036', 22, 3),
('20230036', 28, 6),
('20230037', 4, 6),
('20230037', 10, 6),
('20230037', 16, 7),
('20230037', 22, 4),
('20230037', 28, 8),
('20230038', 5, 10),
('20230038', 11, 10),
('20230038', 17, 10),
('20230038', 23, 10),
('20230038', 29, 10),
('20230039', 5, 5),
('20230039', 11, 8),
('20230039', 17, 8),
('20230039', 23, 9),
('20230039', 29, 10),
('20230040', 5, 5),
('20230040', 11, 7),
('20230040', 17, 8),
('20230040', 23, 8),
('20230040', 29, 4),
('20230041', 5, 10),
('20230041', 11, 10),
('20230041', 17, 9),

```
('20230041', 23, 9),
('20230041', 29, 10),
('20230042', 6, 10),
('20230042', 12, 4),
('20230042', 18, 4),
('20230042', 24, 3),
('20230042', 30, 6),
('20230043', 6, 8),
('20230043', 12, 5),
('20230043', 18, 3),
('20230043', 24, 7),
('20230043', 30, 3),
('20230044', 6, 9),
('20230044', 12, 10),
('20230044', 18, 9),
('20230044', 24, 6),
('20230044', 30, 7),
('20230045', 6, 5),
('20230045', 12, 7),
('20230045', 18, 9),
('20230045', 24, 7),
('20230045', 30, 5);
```

Povoamento da tabela “bimestredealuno”:

```
INSERT INTO bimestredealuno
VALUES
('20230021', 20231, 'Aprovado'),
('20230022', 20231, 'Aprovado'),
('20230024', 20231, 'Aprovado'),
('20230025', 20231, 'Aprovado'),
('20230026', 20231, 'Aprovado'),
('20230027', 20231, 'Reprovado'),
('20230028', 20231, 'Aprovado'),
('20230029', 20231, 'Aprovado'),
('20230030', 20231, 'Aprovado'),
('20230031', 20231, 'Aprovado'),
('20230032', 20231, 'Aprovado'),
('20230033', 20231, 'Aprovado'),
('20230034', 20231, 'Aprovado'),
('20230035', 20231, 'Aprovado'),
('20230036', 20231, 'Transferido'),
('20230037', 20231, 'Aprovado'),
('20230038', 20231, 'Aprovado'),
('20230039', 20231, 'Aprovado'),
('20230040', 20231, 'Transferido'),
('20230041', 20231, 'Expulso'),
('20230042', 20231, 'Transferido'),
('20230043', 20231, 'Expulso'),
('20230044', 20231, 'Transferido'),
('20230045', 20231, 'Aprovado');
```

Povoamento da tabela “professoresporturma”:

```
INSERT INTO professoresporturma
VALUES
('20210001', 1, 2),
('20210001', 2, 2),
('20210001', 3, 2),
('20210001', 4, 2),
('20210001', 5, 2),
('20210001', 6, 2),
('20210002', 1, 1),
('20210002', 2, 1),
('20210002', 3, 1),
('20210002', 4, 1),
('20210002', 5, 1),
('20210002', 6, 1),
('20200003', 1, 5),
('20200003', 2, 5),
('20200003', 3, 5),
('20200003', 4, 5),
('20200003', 5, 5),
('20200003', 6, 5),
('20180004', 1, 3),
('20180004', 2, 3),
('20180004', 3, 3),
('20180004', 4, 3),
('20180004', 5, 3),
('20180004', 6, 3),
('20190005', 1, 4),
('20190005', 2, 4),
('20190005', 3, 4),
('20190005', 4, 4),
('20190005', 5, 4),
('20190005', 6, 4);
```

2.5 - Índices

Índices criados para cada uma das tabelas:

```
CREATE INDEX IndiceProfessor ON Professores(Matricula);
CREATE INDEX IndiceEnderecoProfessor ON EnderecoProfessor(MatriculaProfessor,
Municipio);
CREATE INDEX IndiceSecretarios ON Secretarios(Matricula);
CREATE INDEX IndiceEderecoSecretario ON EnderecoSecretario(MatriculaSecretario,
Municipio);
CREATE INDEX IndiceDisciplinas ON Disciplinas(idDisciplina);
CREATE INDEX IndiceTurmas ON Turmas(idTurma);
CREATE INDEX IndiceAulas ON Aulas(idAula);
CREATE INDEX IndiceAlunos ON Alunos(Matricula);
CREATE INDEX IndiceEnderecoAluno ON EnderecoAlunos(MatriculaAluno, Municipio);
CREATE INDEX IndiceAvaliacoes ON Avaliacoes(idAvaliacao);
CREATE INDEX IndiceBimestres ON Bimestres(idBimestre);
CREATE INDEX IndiceFaltas ON Faltas(idFalta);
CREATE INDEX IndiceFaltasPorAula ON FaltasPorAula(idFalta, idAula);
```

```
CREATE INDEX IndiceAvaliacaoDeAluno ON AvaliacaoDeAluno(MatriculaAluno);
CREATE INDEX IndiceBimestreDeAluno ON BimestreDeAluno(MatriculaAluno);
CREATE INDEX IndiceTurmasDeAluno ON TurmasDeAluno(idTurma, MatriculaAluno);
CREATE INDEX IndiceProfessoresPorTurma ON
ProfessoresPorTurma(MatriculaProfessor);
```

2.6 - Visões

Visão que auxilia na recuperação da matrícula e nome dos alunos que foram aprovados:

```
CREATE VIEW AlunosAprovados AS(
SELECT BA.Matriculaaluno, A.Nome
FROM BimestreDeAluno BA INNER JOIN Alunos A ON BA.matriculaaluno = A.matricula
WHERE BA.Situacao = 'Aprovado');

SELECT * FROM AlunosAprovados;
```

Visão que auxilia na recuperação da matrícula e nome dos professores e o id da turma e a série ministrada por cada um dos professores:

```
CREATE VIEW TurmasProfessor AS
SELECT P.Matricula as MatriculaProfessor, P.Nome as Professor, T.idTurma, T.Serie
FROM (Professores P INNER JOIN ProfessoresPorTurma PPT ON P.Matricula =
PPT.MatriculaProfessor) NATURAL INNER JOIN Turmas T
ORDER BY T.idTurma;

SELECT * FROM TurmasProfessor;
```

Visão que auxilia na recuperação da matrícula e nome dos alunos e o id da turma e a série desses alunos, a fim de saber as turmas atuais de cada aluno:

```
CREATE VIEW TurmasAtuaisDosAlunos AS
SELECT A.Matricula, A.Nome AS NomeAluno, T.idTurma, T.Serie
FROM (Alunos A INNER JOIN TurmasDeAluno TA ON A.Matricula = TA.MatriculaAluno)
NATURAL INNER JOIN Turmas T;

SELECT * FROM TurmasAtuaisDosAlunos;
```

Visão que auxilia na recuperação da matrícula e nome dos secretários e a matrícula e o nome dos alunos matriculados por cada secretário:

```
CREATE VIEW AlunosMatriculadosPorSecretarios AS
SELECT S.Matricula as MatriculaSecretario, S.Nome as Secretario, A.Matricula as
MatriculaAluno, A.Nome as Aluno
FROM Secretarios S INNER JOIN Alunos A ON S.Matricula = A.MatriculaSecretario
ORDER BY S.Matricula;

SELECT * FROM AlunosMatriculadosPorSecretarios;
```

2.7 - Gatilhos

Gatilho para verificar se a faixa salarial dos professores estará em conformidade com a titulação, executado a cada inserção ou atualização na tabela Professores:

```
CREATE FUNCTION verificarSalario()
 RETURNS TRIGGER AS
 $$
 BEGIN
        IF NEW.Titulacao = 'Graduação' AND NEW.Salario >= 1200 AND NEW.Salario <
3500 THEN RETURN NEW;
                ELSE RETURN NULL;
        END IF;
        IF NEW.Titulacao = 'Mestrado' AND NEW.Salario >=3500 AND NEW.Salario <
5000 THEN RETURN NEW;
                ELSE RETURN NULL;
        END IF;
        IF NEW.Titulacao = 'Mestrado' AND NEW.Salario >= 5000 THEN RETURN NEW;
                ELSE RETURN NULL;
        END IF;
 END
 $$
 LANGUAGE PLPGSQL

CREATE TRIGGER verificarSalarioTG
BEFORE INSERT OR UPDATE ON Professores
FOR EACH ROW
EXECUTE PROCEDURE verificarSalario();
```

Gatilho para garantir que a carga horária de cada disciplina estará dentro das especificações, dado que TotalAulas é a carga horária semanal e em um ano há na média 40 semanas de aula, também assumindo que dessas 40 o mínimo para preencher a carga anual é de 30 semanas, a coluna CargaHoraria deve ser equivalente a no mínimo 30 vezes a carga semanal de aulas:

```
CREATE FUNCTION verificarTotalAulas()
 RETURNS TRIGGER AS
 $$
 BEGIN
        IF NEW.Totalaulas < 1 THEN RETURN NULL;
                ELSE IF NEW.Cargahoraria <= NEW.Totalaulas * 30 THEN RETURN
NULL;
                        ELSE RETURN NEW;
                        END IF;
        END IF;
 END
 $$
 LANGUAGE PLPGSQL;

 CREATE TRIGGER verificarTotalAulasTG
 BEFORE INSERT OR UPDATE ON Disciplinas
 FOR EACH ROW
 EXECUTE PROCEDURE verificarTotalAulas();
```

Gatilho para modificar a situação do aluno (aprovado ou reprovado) quando houver uma modificação nas suas notas:

```
CREATE FUNCTION verificarAprovacao() RETURNS TRIGGER AS
$$
DECLARE
        mediaGlobal avaliacaoDeAluno.nota%TYPE;
BEGIN
        SELECT INTO mediaGlobal AVG(AL.nota)
        FROM (Avaliacaodealuno AL INNER JOIN Alunos A ON AL.matriculaaluno =
A.matricula)
                  NATURAL INNER JOIN avaliacoes AV
        WHERE A.matricula = NEW.matriculaAluno
        GROUP BY A.matricula;

        IF mediaGlobal >= 7 THEN
                UPDATE bimestreDeAluno B SET situacao = 'Aprovado'
                WHERE B.matriculaAluno = NEW.matriculaAluno;
        ELSIF mediaGlobal < 7 THEN
                UPDATE bimestreDeALUNO B SET situacao = 'Reprovado'
                WHERE B.matriculaAluno = NEW.matriculaAluno;
                END IF;
        RETURN NEW;
END
$$
LANGUAGE PLPGSQL;

CREATE TRIGGER alunosAprovados AFTER
INSERT OR UPDATE OR DELETE ON avaliacaoDeAluno
FOR EACH ROW
EXECUTE PROCEDURE verificarAprovacao();
```

2.8 - Procedimentos Armazenados

Procedimento para retornar a quantidade de faltas de um aluno, passando sua matrícula:

```
CREATE FUNCTION contarPresenca(CHARACTER(8)) RETURNS INTEGER AS
$$
DECLARE
        buscarAluno ALIAS FOR $1;
        faltas INTEGER;
BEGIN
        SELECT INTO faltas COUNT(*)
        FROM alunos A INNER JOIN faltas F ON A.matricula = F.matriculaaluno
        WHERE A.matricula = buscarAluno
        GROUP BY A.matricula;
        RETURN faltas;
END
$$
LANGUAGE PLPGSQL;
```

Procedimento para retornar a média geral de resultados em uma avaliação, útil principalmente para medir a dificuldade:

```
CREATE FUNCTION verificarDificuldade(INTEGER) RETURNS FLOAT AS
$$
DECLARE
        buscarid ALIAS FOR $1;
        dificuldade FLOAT;
BEGIN
        SELECT INTO dificuldade AVG(nota)
        FROM avaliacoes AV INNER JOIN avaliacaodealuno AA ON AV.idavaliacao =
AA.idavaliacao
        WHERE AV.idavaliacao = buscarid
        GROUP BY AV.idavaliacao;
        RETURN dificuldade;
END
$$
LANGUAGE PLPGSQL;
```

Procedimento para retornar a quantidade de aulas registradas de uma turma, passando identificador da turma em questão:

```
CREATE FUNCTION contagemAulas(INTEGER) RETURNS INTEGER AS
$$
DECLARE
        buscarturma ALIAS FOR $1;
        quantidadeaulas INTEGER;
BEGIN
        SELECT INTO quantidadeaulas COUNT(*)
        FROM turmas T INNER JOIN aulas A on T.idturma = A.idturma
        WHERE T.idturma = buscarTurma
        GROUP BY T.idturma;
        RETURN quantidadeaulas;
END
$$
LANGUAGE PLPGSQL;
```

2.9 - Consultas

2.9.1 - Consultas com junções

Consulta para obter as notas de um aluno em uma avaliação específica:no exemplo é usado a matricula ”20230024” do aluno e id da avaliação “7”:

```
SELECT Alunos.Nome AS NomeAluno, Avaliacoes.Tipo, AvaliacaoDeAluno.Nota
FROM Alunos
JOIN AvaliacaoDeAluno ON Alunos.Matricula = AvaliacaoDeAluno.MatriculaAluno
JOIN Avaliacoes ON AvaliacaoDeAluno.idAvaliacao = Avaliacoes.idAvaliacao
WHERE Alunos.Matricula = '20230024' AND Avaliacoes.idAvaliacao = '7';
```

Consulta para listar informações sobre as disciplinas ministradas por um professor em uma turma específica, no exemplo abaixo a matrícula “20190005” e o id da turma “2” é usada como exemplo:

```
SELECT Professores.Nome AS NomeProfessor, Disciplinas.Nome AS NomeDisciplina,
Turmas.Ano, Turmas.Serie
FROM Professores
JOIN ProfessoresPorTurma ON Professores.Matricula =
ProfessoresPorTurma.MatriculaProfessor
JOIN Turmas ON ProfessoresPorTurma.idTurma = Turmas.idTurma
JOIN Disciplinas ON ProfessoresPorTurma.idDisciplina = Disciplinas.idDisciplina
WHERE Professores.Matricula = '20190005' AND Turmas.idTurma = '2';
```

2.9.2 - Consultas envolvendo comparações com valores nulos

Consulta para listar todos os professores que possuem mais de um telefones cadastrados:

```
SELECT matricula, nome, telefone1, telefone2
FROM professores
WHERE telefone1 IS NOT NULL AND telefone2 IS NOT NULL;
```

Consulta para listar todos os professores que não possuam endereço cadastrado:

```
SELECT professores.matricula, professores.nome
FROM professores
LEFT JOIN enderecoprofessor ON professores.matricula =
enderecoprofessor.matriculaprofessor
WHERE enderecoprofessor.matriculaprofessor IS NULL;
```

2.9.3 - Consultas envolvendo busca por substrings

Consulta para buscar por professores de acordo com seus nomes, no exemplo abaixo retorna todos os professores com “Silva” no nome:

```
SELECT matricula, nome
FROM professores
WHERE nome LIKE '%Silva%';
```

Consulta para buscar professores com e-mails que terminem com 'gmail’ (para saber o email de um específico ou mesmo os domínios dos emails):

```
SELECT matricula, nome, email
FROM professores
WHERE email LIKE '%gmail%';
```

2.9.4 - Consultas envolvendo ordenação

Consulta para buscar pelos alunos em ordem crescente de acordo com a ordem alfabética dos nomes, independente da turma:

```
SELECT matricula, nome
FROM alunos
ORDER BY nome ASC;
```

Consulta para buscar pelos professores em ordem decrescente de acordo com os salários:

```
SELECT matricula, nome, salario
FROM professores
ORDER BY salario DESC;
```

2.9.5 - Consultas aninhadas

Consulta das matrículas de alunos utilizando sua rua, para saber alunos que moram próximos, utilizando a “Rua Josival Messyas”:

```
-- Selecione a matricula dos alunos que moram no logradouro 'Rua Josival
Messyas'(Consulta aninhada)
SELECT matriculaaluno
FROM Enderecoalunos
WHERE logradouro IN(
        SELECT logradouro
        FROM enderecoalunos
        WHERE logradouro = 'Rua Josival Messyas'
);
```

Consulta dos nomes dos professores de acordo com sua titulação:

```
SELECT nome
FROM professores
WHERE titulacao IN(
        SELECT titulacao
        FROM professores
        WHERE titulacao = 'Mestrado'
);
```

2.9.6 - Consultas aninhadas correlacionadas

Consulta dos professores que possuem apenas um telefone cadastrado:

```
SELECT matricula
FROM professores
WHERE EXISTS(
        SELECT *
        FROM turmas
        WHERE telefone2 IS NULL
);
```

Consulta das disciplinas que não lançaram nenhuma avaliação:

```
SELECT Nome
```

```
FROM Disciplinas D
WHERE NOT EXISTS (
    SELECT *
    FROM Avaliacoes Av
    WHERE Av.idDisciplina = D.idDisciplina
);
```

2.9.7 - Consultas usando operações de conjunto

Consulta da matrícula dos alunos que possuem faltas:

```
SELECT A.Matricula
FROM Alunos A
INTERSECT
SELECT F.MatriculaAluno
FROM Faltas F;
```

Consulta da matrícula dos alunos que não possuem faltas:

```
SELECT A.Matricula
FROM Alunos A
EXCEPT
SELECT F.MatriculaAluno
FROM Faltas F;
```

2.9.8 - Consultas usando funções agregadas

Consulta para descobrir a média global (de todas as avaliações) de um aluno específico, utilizando sua matrícula (no exemplo foi utilizado a matrícula 20230029):

```
SELECT avg(a.nota)
FROM avaliacaodealuno a, alunos c
WHERE a.matriculaaluno = c.matricula and c.matricula ='20230029';
```

Consulta para retornar o total da folha de pagamento da escola, somando os salários dos professores e secretários.

```
SELECT SUM(sum) AS folhapagamento
FROM (
    SELECT SUM(salario)
        FROM professores
    UNION
    SELECT SUM(salario)
        FROM secretarios
);
```

2.9.9 - Consultas usando agrupamento

Consulta para retornar a média global (todas as matérias) de cada turma:

```
SELECT a.idturma, AVG(nota)
FROM avaliacoes a, avaliacaodealuno b
WHERE a.idavaliacao = b.idavaliacao
GROUP BY a.idturma;
```

Consulta para retornar a média global apenas das turmas que obtiveram um total acima de 7:

```
SELECT a.idturma, AVG(nota) AS mediaturma
FROM avaliacoes a, avaliacaodealuno b
WHERE a.idavaliacao = b.idavaliacao
GROUP BY a.idturma
HAVING AVG(nota) > 7;
```