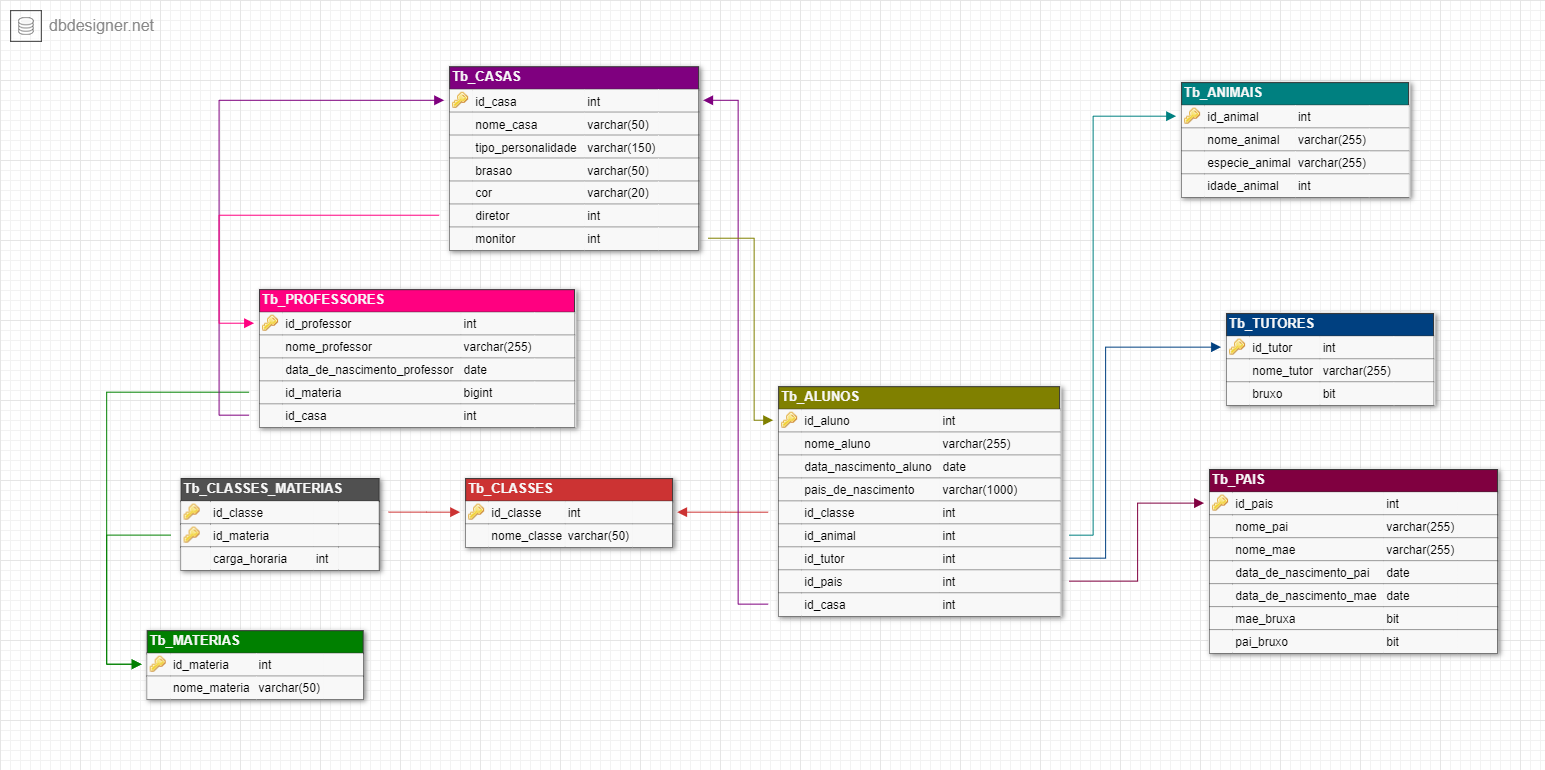
**OBJETIVO DO PROJETO**

Temos como objetivo desenvolver o modelo físico de um banco de dados da Escola de Magia e Bruxaria de Hogwarts.

**ALUNOS**

1. Douglas Fonseca (douglas.barbosa.fonseca@gmail.com)
2. Farley Salomão Filho (farleysalf@gmail.com)
3. Hallynny Barros (hallynnyhenrique01@hotmail.com)
4. Samira Vidal de Lima (samira.vidal1@gmail.com)
5. Thalles Correia ([thallesrafael2010@gmail.com](mailto:thallesrafael2010@gmail.com))
6. Vinicius Maciel e Souza ([v188201@dac.unicamp.br](mailto:v188201@dac.unicamp.br))
7. **Modelagem de Dados**

**DIAGRAMA DE MODELO LÓGICO DOS DADOS**

****

1. **CRIAÇÃO DA BASE DE DADOS – SQL**

CREATE DATABASE HOGWARTS;

1. **CRIAÇÃO DAS TABELAS – SQL**

USE HOGWARTS

CREATE TABLE Tb\_CLASSES(

id\_classe INT IDENTITY(1000,1) NOT NULL,

nome\_classe varchar(50) NOT NULL,

PRIMARY KEY (id\_classe)

);

CREATE TABLE Tb\_MATERIAS(

id\_materia INT IDENTITY(2000,1) NOT NULL,

nome\_materia varchar(50) NOT NULL,

PRIMARY KEY (id\_materia)

);

CREATE TABLE Tb\_CLASSES\_MATERIAS(

id\_classe INT FOREIGN KEY REFERENCES Tb\_CLASSES(id\_classe),

id\_materia INT FOREIGN KEY REFERENCES Tb\_MATERIAS(id\_materia),

carga\_horaria INT NOT NULL,

PRIMARY KEY(id\_classe, id\_materia)

);

CREATE TABLE Tb\_TUTORES(

id\_tutor INT IDENTITY (3000, 1) NOT NULL,

nome\_tutor varchar(255) NOT NULL,

bruxo BIT NOT NULL,

PRIMARY KEY (id\_tutor)

);

CREATE TABLE Tb\_PROFESSORES(

id\_professor INT IDENTITY (4000, 1) NOT NULL,

nome\_professor VARCHAR(255) NOT NULL,

data\_de\_nascimento\_professor DATE,

PRIMARY KEY (id\_professor),

id\_materia INT FOREIGN KEY REFERENCES Tb\_MATERIAS(id\_materia)

);

CREATE TABLE Tb\_ANIMAIS(

id\_animal INT IDENTITY(5000,1) NOT NULL,

nome\_animal VARCHAR(255) NOT NULL,

especie\_animal VARCHAR(255) NOT NULL,

idade\_animal INT NOT NULL,

PRIMARY KEY (id\_animal)

);

CREATE TABLE Tb\_PAIS(

id\_pais INT IDENTITY(5000,1) NOT NULL,

nome\_pai VARCHAR(255) NOT NULL,

nome\_mae VARCHAR(255) NOT NULL,

data\_de\_nascimento\_pai DATE,

data\_de\_nascimento\_mae DATE,

mae\_bruxa BIT NOT NULL,

pai\_bruxo BIT NOT NULL,

PRIMARY KEY (id\_pais)

);

CREATE TABLE Tb\_ALUNOS(

id\_aluno INT IDENTITY(6000,1) NOT NULL,

nome\_aluno VARCHAR(255) NOT NULL,

data\_nascimento\_aluno DATE NOT NULL,

endereco VARCHAR(1000) NOT NULL,

id\_classe INT FOREIGN KEY REFERENCES Tb\_CLASSES(id\_classe),

id\_animal INT FOREIGN KEY REFERENCES Tb\_ANIMAIS(id\_animal),

id\_tutor INT FOREIGN KEY REFERENCES Tb\_TUTORES(id\_tutor),

id\_pais INT FOREIGN KEY REFERENCES Tb\_PAIS(id\_pais)

PRIMARY KEY (id\_aluno)

);

CREATE TABLE Tb\_CASAS(

id\_casa int IDENTITY(1,1) NOT NULL,

nome\_casa varchar(50) NOT NULL,

tipo\_personalidade varchar(150) NOT NULL,

brasao varchar(50) NOT NULL,

cor varchar(20) NOT NULL,

diretor int FOREIGN KEY REFERENCES Tb\_PROFESSORES(id\_professor),

monitor int FOREIGN KEY REFERENCES Tb\_ALUNOS(id\_aluno),

PRIMARY KEY (id\_casa)

);

ALTER TABLE Tb\_PROFESSORES

ADD id\_casa INT FOREIGN KEY REFERENCES Tb\_CASAS(id\_casa);

ALTER TABLE Tb\_ALUNOS

ADD id\_casa INT FOREIGN KEY REFERENCES Tb\_CASAS(id\_casa);

CREATE TABLE Tb\_ALUNOS\_NOTAS(

id\_aluno INT FOREIGN KEY REFERENCES Tb\_alunos(id\_aluno),

id\_materia INT FOREIGN KEY REFERENCES Tb\_MATERIAS(id\_materia),

nota FLOAT,

PRIMARY KEY(id\_aluno, id\_materia)

);

1. **INSERÇÃO DOS DADOS – SQL**

USE HOGWARTS

INSERT INTO Tb\_TUTORES(nome\_tutor,bruxo)

VALUES ('Válter Dursley', 0)

INSERT INTO Tb\_PAIS(nome\_pai, nome\_mae, data\_de\_nascimento\_mae, data\_de\_nascimento\_pai, mae\_bruxa, pai\_bruxo)

VALUES ('Tiago Potter', 'Lílian Evans Potter', '1960-01-30', '1960-03-27', 1, 1),

('Xenofílio Lovegood', 'Pandora Lovegood', '1960-02-16', '1964-02-03', 1, 1),

('Amos Diggory', 'Taís Diggory', '1960-09-01', '1960-09-30', 1, 1),

('Lucius Malfoy', 'Narcissa Malfoy', '1955-01-01', '1953-09-09', 1, 1),

('Wendell Granger', 'Monica Granger', '1959-07-05', '1960-03-17', 0, 0),

('Colm Abbott', 'Adda Abbott', '1952-05-05', '1954-07-18', 0, 1),

('Hari Patil', 'Kali Patil', '1954-11-03', '1955-02-16', 1, 1),

('Bartley Parkinson', 'Aileen Parkinson', '1955-10-19', '1955-12-10', 1, 1)

INSERT INTO Tb\_ANIMAIS(nome\_animal,especie\_animal,idade\_animal)

VALUES ('Edwiges','Coruja', 8),

('Pigwidgeon','Coruja', 7),

('Lu','Coruja',9),

('Pen','Rato',5),

('Bichento','Gato', 5),

('Ott','Coruja', 7),

('Powk','Gato', 4),

('Owl','Coruja', 8)

INSERT INTO Tb\_MATERIAS(nome\_materia)

VALUES ('Tranfiguração'),

('Herbologia'),

('Feitiços'),

('Porções')

INSERT INTO Tb\_CLASSES(nome\_classe)

VALUES ('1 ano'),

('2 ano'),

('3 ano'),

('4 ano')

ALTER TABLE Tb\_CLASSES\_MATERIAS NOCHECK CONSTRAINT ALL

INSERT INTO Tb\_CLASSES\_MATERIAS(id\_classe, id\_materia, carga\_horaria)

VALUES (1000,2000,70),

(1000,2001,60),

(1000,2002,80),

(1000,2003,100),

(1001,2000,60),

(1001,2001,60),

(1001,2002,75),

(1001,2003,85),

(1002,2000,90),

(1002,2001,50),

(1002,2002,65),

(1002,2003,75),

(1003,2000,80),

(1003,2001,90),

(1003,2002,60),

(1003,2003,60)

ALTER TABLE Tb\_CLASSES\_MATERIAS CHECK CONSTRAINT ALL

INSERT INTO Tb\_PROFESSORES (nome\_professor, data\_de\_nascimento\_professor)

VALUES ('Minerva McGonagall', '1935-10-04'),

('Pomona Sprout', '1931-05-15'),

('Filius Flitwick', '1958-10-17'),

('Severo Snape', '1960-09-01');

INSERT INTO Tb\_CASAS(nome\_casa,tipo\_personalidade, brasao, cor, diretor, monitor)

VALUES ('Grifinória', 'coragem, ousadia, determinação, audácia e atrevimento', 'Leão', 'Vermelho e Dourado', NULL, NULL ),

('Lufa-Lufa', 'leais, trabalhadores, pacientes, justos, dedicados e verdadeiros', 'Texugo', 'Amarelo e o Preto', NULL, NULL ),

('Corvinal', 'inteligentes, criativos, perspicazes, prudentes e estudiosos', 'Águia', 'Azul e o Bronze', NULL, NULL),

('Sonserina', 'ambição, astúcia, liderança, desembaraço e individualismo', 'Serpente', 'Verde e o Prateado', NULL, NULL )

ALTER TABLE Tb\_ALUNOS NOCHECK CONSTRAINT ALL

INSERT INTO Tb\_ALUNOS( nome\_aluno, data\_nascimento\_aluno, endereco, id\_classe, id\_animal, id\_tutor, id\_pais, id\_casa)

VALUES ('Harry Tiago Potter', '1980-07-31', 'Inglaterra', 1001, 5000, 3000, 5000, 1),

('Luna Lovegood', '1981-02-13', 'Grã-Bretanha', 1000, 5001, NULL, 5001, 2),

('Cedrico Diggory', '1977-09-01', 'Gra-Bretanha',1003, 5002, NULL, 5002, 3),

('Draco Lucius Malfoy', '1980-06-05', 'Gra-Bretanha', 1001, 5003, NULL, 5003, 4),

('Hermione Jean Granger', '1979-09-19', 'Inglaterra', 1001, 5004, NULL, 5004, 1),

('Ana Abbott', '1979-09-15', 'Gra-Bretanha', 1002, 5005, NULL, 5005, 2),

('Padma Patil', '1980-04-21', 'Irlanda', 1001, 5006, NULL, 5006, 3),

('Pansy Parkinson', '1980-08-31', 'Inglaterra', 1001, 5007, NULL, 5007, 4);

ALTER TABLE Tb\_ALUNOS NOCHECK CONSTRAINT ALL

UPDATE Tb\_CASAS

SET diretor = 4000, monitor = 6004

WHERE id\_casa = 1;

UPDATE Tb\_CASAS

SET diretor = 4001, monitor = 6005

WHERE id\_casa = 2;

UPDATE Tb\_CASAS

SET diretor = 4002, monitor = 6006

WHERE id\_casa = 3;

UPDATE Tb\_CASAS

SET diretor = 4003, monitor = 6007

WHERE id\_casa = 4;

ALTER TABLE Tb\_PROFESSORES NOCHECK CONSTRAINT ALL

UPDATE Tb\_PROFESSORES

SET ID\_CASA = CASE

WHEN ID\_PROFESSOR = 4000 THEN 1

WHEN ID\_PROFESSOR = 4001 THEN 2

WHEN ID\_PROFESSOR = 4002 THEN 3

WHEN ID\_PROFESSOR = 4003 THEN 4

END

UPDATE Tb\_PROFESSORES

SET ID\_MATERIA = CASE

WHEN ID\_PROFESSOR = 4000 THEN 2000

WHEN ID\_PROFESSOR = 4001 THEN 2001

WHEN ID\_PROFESSOR = 4002 THEN 2002

WHEN ID\_PROFESSOR = 4003 THEN 2003

END

ALTER TABLE Tb\_PROFESSORES CHECK CONSTRAINT ALL

INSERT INTO Tb\_ALUNOS\_NOTAS (id\_aluno, id\_materia)

SELECT A.ID\_ALUNO, B.ID\_MATERIA FROM

Tb\_ALUNOS A CROSS JOIN Tb\_MATERIAS B

ORDER BY A.id\_aluno ASC;

UPDATE Tb\_ALUNOS\_NOTAS

SET nota = abs(checksum(NewId()) % 10) + 2

UPDATE Tb\_ALUNOS\_NOTAS

SET nota = CASE

WHEN nota > 10 THEN 10

ELSE nota

END;

UPDATE Tb\_ALUNOS\_NOTAS

SET nota = CASE

WHEN ID\_ALUNO = 6004 THEN 10

ELSE nota

END;

1. **CONSULTAS – SQL**

-- 1) Média de notas por aluno

SELECT A.nome\_aluno AS Aluno, AVG(B.nota) AS [Nota Média] FROM

Tb\_ALUNOS A INNER JOIN Tb\_ALUNOS\_NOTAS B

ON A.id\_aluno = B.id\_aluno

GROUP BY A.NOME\_ALUNO

ORDER BY AVG(B.nota) DESC;

-- 2) Média das notas por casa

SELECT D.nome\_casa AS Casa, AVG(B.NOTA) AS [Nota Média] FROM

Tb\_ALUNOS A INNER JOIN Tb\_ALUNOS\_NOTAS B

ON A.id\_aluno = B.id\_aluno

INNER JOIN

[Tb\_CASAS] D

ON A.id\_casa = D.id\_casa

GROUP BY D.nome\_casa;

-- 3) Distribuição do tipo de sangue dos alunos

SELECT

C.nome\_casa AS [Casa],

SUM(CASE

WHEN (CONVERT(INT, B.mae\_bruxa) + CONVERT(INT, B.pai\_bruxo)) < 2 THEN 0

WHEN (CONVERT(INT, B.mae\_bruxa) + CONVERT(INT, B.pai\_bruxo)) = 2 THEN 1

end)

AS [Sangue puro],

SUM(CASE

WHEN (CONVERT(INT, B.mae\_bruxa) + CONVERT(INT, B.pai\_bruxo)) < 2 THEN 1

WHEN (CONVERT(INT, B.mae\_bruxa) + CONVERT(INT, B.pai\_bruxo)) = 2 THEN 0

end)

AS [Mestiço]

FROM

Tb\_ALUNOS A INNER JOIN Tb\_PAIS B

ON A.id\_pais = B.id\_pais

INNER JOIN Tb\_CASAS C

ON A.id\_casa = C.id\_casa

GROUP BY C.nome\_casa

ORDER BY [Casa];

-- 4) Professor mais velho de cada casa

SELECT tc.nome\_casa AS Casa, tp.nome\_professor AS Professor, DATEDIFF(year, data\_de\_nascimento\_professor, GETDATE()) AS Idade

FROM Tb\_CASAS tc inner join Tb\_PROFESSORES tp

on tc.id\_casa = tp.id\_casa

ORDER BY tc.nome\_casa;

-- 5) Nota média por matéria

SELECT A.nome\_materia AS [Matéria], ROUND(AVG(B.nota), 2) AS [Média de Notas] FROM

Tb\_MATERIAS A INNER JOIN Tb\_ALUNOS\_NOTAS B

ON A.id\_materia = B.id\_materia

GROUP BY A.nome\_materia

ORDER BY [Média de Notas] DESC;

-- 6) Número de Alunos por casa

SELECT A.nome\_casa AS Casa, COUNT(\*) AS [Número de Alunos] FROM

Tb\_CASAS A INNER JOIN Tb\_ALUNOS B

ON A.id\_casa = B.id\_casa

GROUP BY A.nome\_casa;

- usar um select case e subselects para criar uma view para o projeto

--7) Nota dos alunos por matéria / (Aprovado/Reprovado)

SELECT A.nome\_aluno AS Aluno, C.nome\_materia AS Matéria, AVG(B.nota) AS [Nota Média], (CASE

WHEN (AVG(B.nota)) >= 6 THEN 'Aprovado'

ELSE 'Reprovado'

END)

AS [Aprovado] FROM

Tb\_ALUNOS A INNER JOIN Tb\_ALUNOS\_NOTAS B

ON A.id\_aluno = B.id\_aluno

INNER JOIN Tb\_MATERIAS C

ON B.id\_materia = C.id\_materia

GROUP BY C.nome\_materia, A.nome\_aluno

ORDER BY A.nome\_aluno, C.nome\_materia DESC;

- usar estrutura de repetição na criação de function ou procedure

--8) Procedure (Pesquisar notas de um aluno por matéria)

CREATE PROCEDURE Pc\_NOTAS\_ALUNO

@NomeAluno VARCHAR (255)

AS

SELECT A.nome\_aluno AS Aluno, C.nome\_materia AS Matéria, AVG(B.nota) AS [Nota Média], (CASE

WHEN (AVG(B.nota)) >= 6 THEN 'Aprovado'

ELSE 'Reprovado'

END)

AS [Aprovado] FROM

Tb\_ALUNOS A INNER JOIN Tb\_ALUNOS\_NOTAS B

ON A.id\_aluno = B.id\_aluno

INNER JOIN Tb\_MATERIAS C

ON B.id\_materia = C.id\_materia

WHERE A.nome\_aluno = @NomeAluno

GROUP BY C.nome\_materia, A.nome\_aluno

ORDER BY A.nome\_aluno, C.nome\_materia DESC

- usar estrutura condicional na criação de trigger

--9) Trigger para criar todas as combinações de uma nova matéria com as 7 classes na tabela Tb\_CLASSES\_MATERIAS

CREATE TRIGGER TGR\_COMBINA\_MAT\_CLASS

ON Tb\_MATERIAS

AFTER INSERT

AS

BEGIN

INSERT INTO Tb\_CLASSES\_MATERIAS (id\_materia, id\_classe, carga\_horaria)

SELECT A.id\_materia, B.id\_classe, 60

FROM Tb\_MATERIAS A CROSS JOIN Tb\_CLASSES B

WHERE id\_materia=(SELECT max(id\_materia) FROM Tb\_MATERIAS)

END;

10) Procedure para atualizar a carga horária da matéria.

CREATE PROCEDURE Pc\_MOD\_CH

@Classe INT, @Mat INT, @CH INT

AS

UPDATE Tb\_CLASSES\_MATERIAS

SET carga\_horaria = @CH

WHERE id\_classe = @Classe AND id\_materia = @Mat

Sugestão: Um aluno disse que somente sangues puros são inteligentes, prove o contrário mostrando os dados da média dos alunos por tipo sanguineo junto à média por casa . (dica Union)

SELECT

CASE

WHEN (CONVERT(INT, B.mae\_bruxa) + CONVERT(INT, B.pai\_bruxo)) < 2 THEN 'Mestiço'

WHEN (CONVERT(INT, B.mae\_bruxa) + CONVERT(INT, B.pai\_bruxo)) = 2 THEN 'Sangue Puro'

end

AS [Sangue],

ROUND(AVG(C.nota), 2) AS [Nota Média],

ROUND(STDEV(C.nota), 2) AS [Desvio Padrão]

FROM

Tb\_ALUNOS A INNER JOIN Tb\_PAIS B

ON A.id\_pais = B.id\_pais

INNER JOIN Tb\_ALUNOS\_NOTAS C

ON A.id\_aluno = C.id\_aluno

GROUP BY

CASE

WHEN (CONVERT(INT, B.mae\_bruxa) + CONVERT(INT, B.pai\_bruxo)) < 2 THEN 'Mestiço'

WHEN (CONVERT(INT, B.mae\_bruxa) + CONVERT(INT, B.pai\_bruxo)) = 2 THEN 'Sangue Puro'

end

