**UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ**
**DEPARTAMENTO DE INFORMÁTICA**
**BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO**

**WALDYR TURQUETTI GONÇALVES**

# FUNDAÇÃO PRO SANGUE

**TRABALHO ACADÊMICO**

**PONTA GROSSA**
**2019**

**WALDYR TURQUETTI GONÇALVES**

# FUNDAÇÃO PRO SANGUE

Trabalho acadêmico apresentado como requisito parcial à aprovação na disciplina de Banco de Dados I do Departamento de Informática da Universidade Tecnológica Federal do Paraná.

Professor responsável:
Profª. Drª. Simone de Almeida

**PONTA GROSSA**
**2019**

**SUMÁRIO**

# 1. INTRODUÇÃO

A Fundação Pro-Sangue planeja desenvolver um sistema para gerenciar as suas tarefas diárias. O processo de doação passa por algumas etapas descritas, que a base de dados deve considerar:

- O candidato à doação informa na recepção da Fundação Pro-Sangue seus dados pessoais como: Nome, sexo, endereço completo, data de nascimento, nome dos pais e apresenta o documento de identidade original, caso não seja doador frequente. Caso o candidato já esteja cadastrado, seus dados são confirmados pelo atendente. Um código é gerado pelo sistema para rastreamento do doador, doação e exames realizados assim como seus resultados, a data e horário da doação.
- Em caso do doador frequente, o sistema deve verificar a data da última doação, que não deve ser inferior a 90 dias se o candidato for do sexo masculino ou 120 dias se feminino.
- O primeiro passo é a Triagem Clínica, onde o candidato responde a uma entrevista com o objetivo de avaliar se a doação pode trazer riscos para ele ou para o receptor. É fundamental responder corretamente às perguntas, caso o candidato a doador possua.
- O passo seguinte é a realização do Teste de Anemia. Este exame é feito para verificar se o candidato à doação possui níveis de hemoglobina dentro do aceitável. Caso não esteja dentro do padrão, o doador é dispensado.
- São coletados aproximadamente 450ml de sangue em uma bolsa de uso único e estéril, sendo, portanto, a coleta de sangue totalmente segura.
- São realizados diversos testes do sangue, cujos resultados devem ser armazenados no banco, como:
    - Triagem sorológica: Hepatite B, Hepatite C, Doença de Chagas, Sífilis, AIDS, HTLV I/II;
    - Imunohematologia: determinação do tipo sanguíneo ABO e Rh, além da pesquisa de anticorpos irregulares.

Os testes descritos são realizados a cada doação, e os resultados serão impressos na Carteirinha do Doador. Caso haja alguma alteração no resultado, o doador será comunicado (a) e talvez seja necessário repetir os exames.

Lembre-se de que esses testes têm o objetivo de triagem e não de diagnóstico, podendo ocorrer resultados falso-positivos. Assim, o eventual resultado positivo para um ou mais testes não deverá ser interpretado como diagnóstico definitivo. Portanto, não há necessidade de preocupação se for convocado (a) para uma consulta médica ou para repetição de exame. Assim, um resultado REAGENTE em um ou mais desses testes pode NÃO ser definitivo, devendo ser analisado em conjunto com a história clínica e outros dados laboratoriais.

Por carta registrada, o doador com algum resultado alterado nos exames laboratoriais é convocado para ser esclarecido e para coletar nova amostra. O esclarecimento é feito por um dos médicos do Banco de Sangue, apenas pessoalmente e de maneira individual. Em algumas situações, o doador recebe uma carta explicativa sobre o resultado alterado, e, caso deseje, poderá agendar consulta médica para esclarecimentos adicionais.

Quando o doador apresenta alteração sorológica que necessita de investigação adicional para esclarecimento diagnóstico, é informado pelo médico do banco de sangue numa segunda consulta e orientado a procurar um especialista de sua preferência.

Os critérios utilizados na triagem clínica e sorológica dos doadores de sangue visam obter o sangue mais seguro possível (que a Medicina atual permite) para uso transfusional.

# 2. DESENVOLVIMENTO

## 2.1. MODELO DE ENTIDADE E RELACIONAMENTO

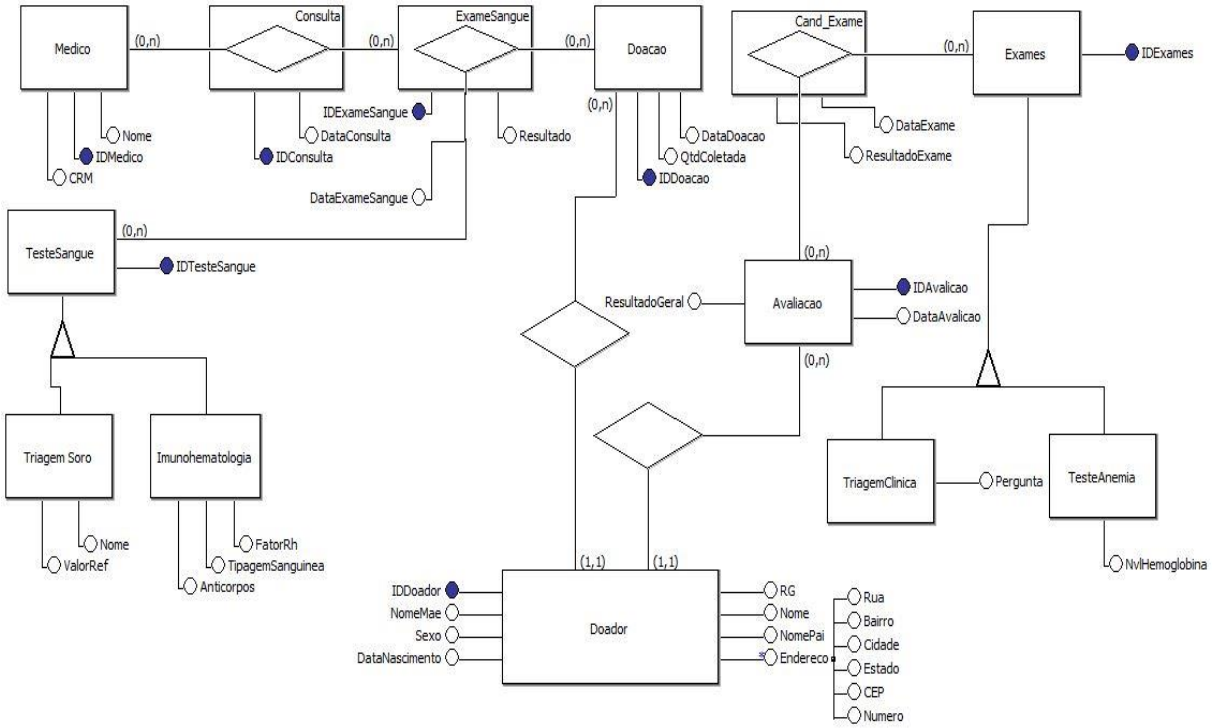


**Figura 1 - MER**

## 2.2. MODELO RELACIONAL

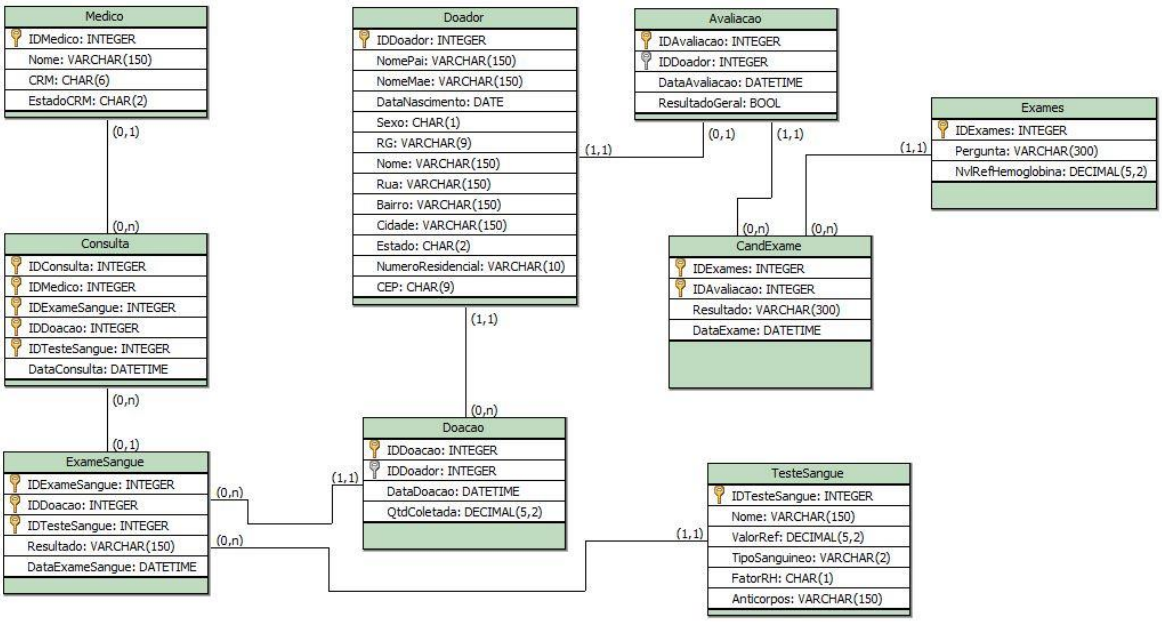


**Figura 2 – MR**

## 2.3 SCRIPT SQL

### 2.3.1 TABELAS, PK E FK

```
CREATE DATABASE FUNDACAO_PROSANGUE;
USE FUNDACAO_PROSANGUE;

CREATE TABLE TesteSangue (
IDTesteSangue INTEGER PRIMARY KEY AUTO_INCREMENT,
Nome VARCHAR(150),
ValorRef DECIMAL(5,2),
TipoSanguineo VARCHAR(2),
FatorRH CHAR(1),
Anticorpos VARCHAR(150)
);

CREATE TABLE Doador (
IDDoador INTEGER PRIMARY KEY AUTO_INCREMENT,
NomePai VARCHAR(150) NOT NULL,
NomeMae VARCHAR(150) NOT NULL,
DataNascimento DATE NOT NULL,
```

```
Sexo CHAR(1) NOT NULL,
RG VARCHAR(9) NOT NULL,
Nome VARCHAR(150) NOT NULL,
Rua VARCHAR(150) NOT NULL,
Bairro VARCHAR(150) NOT NULL,
Cidade VARCHAR(150) NOT NULL,
Estado CHAR(2) NOT NULL,
NumeroResidencial VARCHAR(10) NOT NULL,
CEP CHAR(9) NOT NULL
);

CREATE TABLE Doacao (
IDDoacao INTEGER PRIMARY KEY AUTO_INCREMENT,
IDDoador INTEGER,
DataDoacao DATETIME NOT NULL,
QtdColetada DECIMAL(5,2) DEFAULT 0.00,
FOREIGN KEY(IDDoador) REFERENCES Doador (IDDoador) ON UPDATE RESTRICT ON DELETE
RESTRICT
);
CREATE TABLE ExameSangue (
IDExameSangue INTEGER AUTO_INCREMENT,
IDDoacao INTEGER,
IDTesteSangue INTEGER,
Resultado VARCHAR(300) NOT NULL,
DataExameSangue DATETIME NOT NULL,
PRIMARY KEY(IDExameSangue,IDDoacao,IDTesteSangue),
FOREIGN KEY(IDTesteSangue) REFERENCES TesteSangue (IDTesteSangue) ON UPDATE
RESTRICT ON DELETE RESTRICT,
FOREIGN KEY(IDDoacao) REFERENCES Doacao (IDDoacao) ON UPDATE RESTRICT ON DELETE
RESTRICT
);

CREATE TABLE Medico (
IDMedico INTEGER PRIMARY KEY AUTO_INCREMENT,
Nome VARCHAR(150) NOT NULL,
CRM CHAR(6) NOT NULL,
EstadoCRM CHAR(2) NOT NULL
);

CREATE TABLE Exames (
```

```
IDExames INTEGER PRIMARY KEY AUTO_INCREMENT,
Pergunta VARCHAR(300),
NvlRefHemoglobina DECIMAL(5,2)
);

CREATE TABLE Avaliacao (
IDAvaliacao INTEGER PRIMARY KEY AUTO_INCREMENT,
IDDoador INTEGER,
DataAvaliacao DATETIME NOT NULL,
ResultadoGeral BOOL,
FOREIGN KEY(IDDoador) REFERENCES Doador (IDDoador) ON UPDATE RESTRICT ON DELETE RESTRICT
);

CREATE TABLE CandExame (
IDExames INTEGER,
IDAvaliacao INTEGER,
Resultado VARCHAR(300) NOT NULL,
DataExame DATETIME NOT NULL,
PRIMARY KEY(IDExames,IDAvaliacao),
FOREIGN KEY(IDExames) REFERENCES Exames (IDExames) ON UPDATE RESTRICT ON DELETE RESTRICT,
FOREIGN KEY(IDAvaliacao) REFERENCES Avaliacao (IDAvaliacao) ON UPDATE RESTRICT ON DELETE RESTRICT
);

CREATE TABLE Consulta (
IDConsulta INTEGER AUTO_INCREMENT,
IDMedico INTEGER,
IDExameSangue INTEGER,
IDDoacao INTEGER,
IDTesteSangue INTEGER,
DataConsulta DATETIME NOT NULL,
PRIMARY KEY(IDConsulta,IDMedico,IDExameSangue,IDDoacao,IDTesteSangue),
FOREIGN KEY(IDExameSangue,IDDoacao,IDTesteSangue)
REFERENCES ExameSangue (IDExameSangue,IDDoacao,IDTesteSangue) ON UPDATE RESTRICT ON DELETE RESTRICT,
FOREIGN KEY(IDMedico) REFERENCES Medico (IDMedico) ON UPDATE RESTRICT ON DELETE RESTRICT
);
```

### 2.3.2 ÍNDICES

```
CREATE INDEX IDX_DoadorRG on Doador(RG);
CREATE INDEX IDX_DATADOACAO ON Doacao(DataDoacao);
CREATE INDEX IDX_MedicoCRM on Medico(CRM);
CREATE INDEX IDX_MedicoESTADOCRM on Medico(ESTADOCRM);
CREATE INDEX IDX_DATAAVALIACAO ON AVALIACAO(DATAAVALIACAO);
CREATE INDEX IDX_RESULTADOAVALIACAO ON AVALIACAO(RESULTADOGERAL);
CREATE INDEX IDX_DATACONSULTA ON CONSULTA(DATACONSULTA);
CREATE INDEX IDX_DATAEXAME ON CANDEXAME(DATAEXAME);
CREATE INDEX IDX_DATAEXAMESANGUE ON EXAMESANGUE(DATAEXAMESANGUE);
```

### 2.3.3 STOREDE PROCEDURE

```
DELIMITER //
CREATE PROCEDURE SP_INSEREDOADOR
(IN PNOME VARCHAR(150), PNOMEPAI VARCHAR(150), PNOMEMAE VARCHAR(150), PDATANASCIMENTO DATE, PSEXO CHAR(1), PRG CHAR(9), PRUA VARCHAR(150), PBAIRRO VARCHAR(150), PCIDADE VARCHAR(150), PESTADO CHAR(2), PNUMERORESIDENCIA VARCHAR(10), PCEP CHAR(9), OUT MENSAGEM VARCHAR(100))
BEGIN
	IF(NOT EXISTS(SELECT RG
			FROM DOADOR
			WHERE PRG = RG)) THEN
	BEGIN
		INSERT INTO DOADOR (NOME, NOMEPAI, NOMEMAE, DATANASCIMENTO, SEXO, RG, RUA, BAIRRO, CIDADE, ESTADO, NUMERORESIDENCIAL, CEP)
		VALUES (PNOME, PNOMEPAI, PNOMEMAE, PDATANASCIMENTO, PSEXO, PRG, PRUA, PBAIRRO, PCIDADE, PESTADO, PNUMERORESIDENCIA, PCEP);

		SET MENSAGEM = 'Candidato inserido no Banco de Dados, Operação realizada com sucesso!';
	END;
	ELSE

		SET MENSAGEM = 'Candidato existente no Banco de Dados, operação cancelada!';

	END IF;

END//
DELIMITER ;
```

```
DELIMITER //
CREATE PROCEDURE SP_ALTERADOADOR (IN PCODIGO INTEGER,IN PNOME VARCHAR(150),
PNOMEPAI VARCHAR(150), PNOMEMAE VARCHAR(150), PDATANASCIMENTO DATE, PSEXO
CHAR(1), PRG CHAR(9), PRUA VARCHAR(150), PBAIRRO VARCHAR(150), PCIDADE
VARCHAR(150), PESTADO CHAR(2), PNUMERORESIDENCIA VARCHAR(10), PCEP CHAR(9),
OUT MENSAGEM VARCHAR(100))
BEGIN
        IF (NOT EXISTS (SELECT IDDOADOR
                        FROM DOADOR
                            WHERE IDDOADOR <> PCODIGO AND RG = PRG)) THEN
        BEGIN
                UPDATE DOADOR
                SET NOME = PNOME,
                NOMEPAI = PNOMEPAI,
                NOMEMAE = PNOMEMAE,
                DATANASCIMENTO = PDATANASCIMENTO,
                SEXO = PSEXO,
                RG = PRG,
                RUA = PRUA,
                BAIRRO = PBAIRRO,
                CIDADE = PCIDADE,
                ESTADO = PESTADO,
                NUMERORESIDENCIAL = PNUMERORESIDENCIA,
                CEP = PCEP
                WHERE IDDoador = PCODIGO;

                SET MENSAGEM = 'Operação realizada com sucesso!';
        END;
        ELSE
                SET MENSAGEM = 'RG incorreto, operação não realizada!';
        END IF;
END//
DELIMITER ;


DELIMITER //
CREATE PROCEDURE SP_INSEREMEDICO
(IN PNOME VARCHAR(150),IN PCODIGO CHAR(6), IN PESTADOCRM CHAR(2), OUT MENSAGEM
VARCHAR(100))
```

```
BEGIN
	IF(NOT EXISTS(SELECT IDMEDICO FROM MEDICO WHERE NOME = PNOME AND CRM =
PCODIGO AND ESTADOCRM = PESTADOCRM)) THEN
	BEGIN
		INSERT INTO MEDICO (NOME,CRM,ESTADOCRM)
		VALUES (PNOME,PCODIGO,PESTADOCRM);

		SET MENSAGEM = 'Operação realizada com sucesso!';
	END;
	ELSE
		SET MENSAGEM = 'Medico existente, operação cancelada!';
	END IF;

END//
DELIMITER ;


DELIMITER //
CREATE PROCEDURE SP_ALTERAMEDICO (IN PNOME VARCHAR(150), IN PCODIGO INTEGER,
IN PCRM CHAR(6), IN PESTADOCRM CHAR(2), OUT MENSAGEM VARCHAR(100))
BEGIN
	IF (NOT EXISTS (SELECT IDMEDICO
			FROM MEDICO
			   WHERE IDMEDICO <> PCODIGO AND CRM = PCRM AND PESTADOCRM
= ESTADOCRM)) THEN
	BEGIN
		UPDATE MEDICO
		SET NOME = PNOME,
		CRM = PCRM,
		ESTADOCRM = PESTADOCRM
		WHERE IDMEDICO = PCODIGO;

		SET MENSAGEM = 'Operação realizada com sucesso!';
	END;
	ELSE
		SET MENSAGEM = 'CRM incorreto, operação não realizada!';
	END IF;
END//
DELIMITER ;
```

```
DELIMITER //
CREATE PROCEDURE SP_INSEREDOACAO
(IN PCODIGO INTEGER, PDATADOACAO DATETIME, PQTDCOLETADA DECIMAL(5,2), OUT MENSAGEM VARCHAR(100))
BEGIN
	DECLARE AUX1 SMALLINT;
	SET AUX1 = (SELECT (DATEDIFF(CURDATE(), MAX( CAST(D.DATADOACAO AS DATE) ))) AS ULTDOACAO
	FROM DOADOR AS DR, DOACAO AS D
	WHERE  PCODIGO = D.IDDOADOR);

	IF((( AUX1 > 90 AND (SELECT SEXO FROM  DOADOR WHERE IDDOADOR = PCODIGO) = 'M') OR
	( AUX1 > 120 AND (SELECT SEXO FROM  DOADOR WHERE IDDOADOR = PCODIGO) = 'F') OR AUX1 IS NULL) AND (SELECT RESULTADOGERAL FROM AVALIACAO
	WHERE IDDOADOR = PCODIGO AND DATAAVALIACAO =
	(SELECT MAX(DATAAVALIACAO) FROM AVALIACAO WHERE IDDOADOR = PCODIGO)) IS TRUE)  THEN

	BEGIN

		INSERT INTO DOACAO (IDDOADOR, DATADOACAO, QTDCOLETADA)
		VALUES (PCODIGO, PDATADOACAO, PQTDCOLETADA);

		SET MENSAGEM = 'Operação realizada com sucesso!';
	END;
	ELSE
		SET MENSAGEM = 'Doador não compriu os requisitos necessários para doar, operação cancelada!';
	END IF;

END//
DELIMITER ;


DELIMITER //
CREATE PROCEDURE SP_ALTERADOACAO
```

```
(IN PCODIGO INTEGER, PDOADOR INTEGER, PDATA DATETIME, PQTDCOLETADA DECIMAL(5,2), OUT MENSAGEM VARCHAR(100))
BEGIN
	IF (NOT EXISTS(SELECT IDDOADOR FROM DOACAO
			WHERE IDDOACAO <> PCODIGO AND
			    DATADOACAO = PDATA AND
			    IDDOADOR = PDOADOR AND QTDCOLETADA = PQTDCOLETADA)) THEN
	BEGIN
		UPDATE DOACAO
		SET DATADOACAO = PDATA, IDDOADOR = PDOADOR,
		QTDCOLETADA = PQTDCOLETADA
		WHERE IDDOACAO = PCODIGO;

		SET MENSAGEM = 'Operação realizada com sucesso!';
	END;
	ELSE
		SET MENSAGEM = 'Doador já realizou uma doação nessa data e horário, alteração não realizada';
	END IF;
END//
DELIMITER ;


DELIMITER //
CREATE PROCEDURE SP_INSEREAVALIACAO
(IN PCODIGO INTEGER, PDATAAVALIACAO DATETIME, PRESULTADOGERAL BOOL, OUT MENSAGEM VARCHAR(100))
BEGIN
	IF(NOT EXISTS(SELECT IDAVALIACAO FROM  AVALIACAO
			WHERE CAST(DATAAVALIACAO AS DATE) = CAST(PDATAAVALIACAO AS DATE) AND
			IDDOADOR = PCODIGO )) THEN

	BEGIN

		INSERT INTO AVALIACAO (IDDOADOR, DATAAVALIACAO, RESULTADOGERAL)
		VALUES (PCODIGO, PDATAAVALIACAO, PRESULTADOGERAL);

			SET MENSAGEM = 'Operação realizada com sucesso!';
```

```
        END;
        ELSE

            SET MENSAGEM = 'Candidato já fez uma avalição no dia, operação cancelada!';
        END IF;

END //
DELIMITER ;


DELIMITER //
CREATE PROCEDURE SP_ALTERAAVALIACAO
(IN PCODIGO INTEGER, PDOADOR INTEGER, PDATAAVALIACAO DATETIME, PRESULTADOGERAL BOOL, OUT MENSAGEM VARCHAR(100))
BEGIN
    IF (NOT EXISTS(SELECT IDAVALIACAO FROM AVALIACAO
                WHERE IDAVALIACAO <> PCODIGO AND
                    PDATAAVALIACAO = DATAAVALIACAO AND
                    IDDOADOR = PDOADOR )) THEN
    BEGIN
        UPDATE AVALIACAO
        SET IDDOADOR = PDOADOR,
        DATAAVALIACAO = PDATAAVALIACAO,
        RESULTADOGERAL = PRESULTADOGERAL
        WHERE IDAVALIACAO = PCODIGO;

        SET MENSAGEM = 'Operação realizada com sucesso!';
    END;
    ELSE
        SET MENSAGEM = 'Candidato já fez avaliação nessa data e horário já cadastrados, alteração não realizada';
    END IF;
END//
DELIMITER ;


DELIMITER //
CREATE PROCEDURE SP_INSEREEXAMES
```

```
(IN PPERGUNTA VARCHAR(300), PNVLREFHEMOGLOBINA DECIMAL(5,2), OUT MENSAGEM VARCHAR(100))
BEGIN
	IF ( (PPERGUNTA IS NOT NULL AND NOT EXISTS( SELECT IDEXAMES
	FROM EXAMES WHERE UPPER(PERGUNTA) = UPPER(PPERGUNTA) ) ) OR
	(NOT EXISTS(SELECT IDEXAMES FROM EXAMES
	WHERE NVLREFHEMOGLOBINA IS NOT NULL))) THEN
	BEGIN
		INSERT INTO EXAMES (PERGUNTA, NVLREFHEMOGLOBINA)
		VALUES (PPERGUNTA, PNVLREFHEMOGLOBINA);

		SET MENSAGEM = 'Operação realizada com sucesso!';
	END;
	ELSE
		SET MENSAGEM = 'Informação de exame já existente e/ou Dados inseridos de forma incorreta, operação cancelada!';
	END IF;

END//
DELIMITER ;


DELIMITER //
CREATE PROCEDURE SP_ALTERAEXAME
(IN PCODIGO INTEGER, IN PPERGUNTA VARCHAR(300), IN PNVLREFHEMOGLOBINA DECIMAL(5,2), OUT MENSAGEM VARCHAR(100))
BEGIN
	IF ( ((NOT EXISTS(SELECT IDEXAMES
	FROM EXAMES WHERE IDEXAMES <> PCODIGO AND UPPER(PERGUNTA) = UPPER(PPERGUNTA) )) AND PNVLREFHEMOGLOBINA IS NULL)  OR
	((NOT EXISTS(SELECT IDEXAMES
	FROM EXAMES)) AND PPERGUNTA IS NULL) ) THEN

	BEGIN
		UPDATE EXAMES
			SET PERGUNTA = PPERGUNTA,
			NVLREFHEMOGLOBINA = PNVLREFHEMOGLOBINA
			WHERE IDEXAMES = PCODIGO;
```

```
		SET MENSAGEM = 'Operação realizada com sucesso!';
	END;
	ELSE
		SET MENSAGEM = 'Informação de exame já existente e/ou Dados inseridos de forma incorreta , operação cancelada!';
	END IF;

END//
DELIMITER ;



DELIMITER //
CREATE PROCEDURE SP_INSERECONSULTA
(IN PCODIGO INTEGER,PDATA DATETIME,PCODIGO2 INTEGER,PCODIGO3 INTEGER,PCODIGO4 INTEGER, OUT MENSAGEM VARCHAR(100))
BEGIN
	IF(NOT EXISTS(SELECT IDCONSULTA
			FROM CONSULTA
			WHERE (PCODIGO=IDMEDICO AND PDATA=DATACONSULTA) OR
				PCODIGO3 = IDDOACAO AND PDATA = DATACONSULTA
)) THEN
	BEGIN
		INSERT INTO CONSULTA (IDMEDICO,DATACONSULTA,IDEXAMESANGUE,IDDOACAO,IDTESTESANGUE)
		VALUES (PCODIGO,PDATA,PCODIGO2,PCODIGO3,PCODIGO4);

		SET MENSAGEM = 'Consulta agendada com sucesso!';
	END;
	ELSE
		SET MENSAGEM = 'Ja existe uma consulta agendada para essa doação, operação cancelada!';
	END IF;

END//
DELIMITER ;


DELIMITER //
CREATE PROCEDURE SP_ALTERACONSULTA
```

```
(IN PIDCONSULTA INTEGER,PIDMEDICO INTEGER,PIDDOACAO INTEGER,PDATACONSULTA DATETIME,OUT MENSAGEM VARCHAR (150))
BEGIN
	IF(NOT EXISTS(SELECT IDCONSULTA
			FROM CONSULTA
			WHERE PIDCONSULTA<>IDCONSULTA AND ( (PIDMEDICO=IDMEDICO AND PDATACONSULTA=DATACONSULTA) OR (PIDDOACAO=IDDOACAO AND PDATACONSULTA=DATACONSULTA) ) )) THEN
	BEGIN
		UPDATE CONSULTA
		SET IDDOACAO=PIDDOACAO,
		   IDMEDICO=PIDMEDICO,
		   DATACONSULTA=PDATACONSULTA
		   WHERE IDCONSULTA=PIDCONSULTA;

		SET MENSAGEM='Data da consulta alterada!';
		END;
		ELSE
			SET MENSAGEM='Ja existe uma consulta com esse medico marcada ou uma consulta para essa doacao para essa data,operação cancelada.';
		END IF;
END//
DELIMITER ;


DELIMITER //
CREATE PROCEDURE SP_INSEREEXAMESANGUE
(IN PCODIGO1 INTEGER,PCODIGO2 INTEGER,PRESULTADO VARCHAR(150),PDATA DATETIME, OUT MENSAGEM VARCHAR(100))
BEGIN
	IF(NOT EXISTS(SELECT IDEXAMESANGUE
			FROM EXAMESANGUE
			WHERE PCODIGO1=IDDOACAO AND
			PCODIGO2=IDTESTESANGUE) ) THEN
	BEGIN
		INSERT INTO EXAMESANGUE (IDDOACAO,IDTESTESANGUE,RESULTADO,DATAEXAMESANGUE)
		VALUES (PCODIGO1,PCODIGO2,PRESULTADO,PDATA);

		SET MENSAGEM = 'Exame de sangue armazenado com sucesso!';
```

```
        END;
        ELSE
                SET MENSAGEM = 'Ja existe um exame de sangue armazenado para essa doação, operação cancelada!';
        END IF;

END//
DELIMITER ;




DELIMITER //
CREATE PROCEDURE SP_ALTERAEXAMESANGUE
(IN PCODEXAMESANGUE INTEGER, PCODDOACAO INTEGER, PCODTESTESANGUE INTEGER, PRESULTADO VARCHAR(150), PDATA DATETIME, OUT MENSAGEM VARCHAR(100))
BEGIN
        IF( NOT EXISTS(SELECT IDEXAMESANGUE
                                        FROM EXAMESANGUE
                                        WHERE IDEXAMESANGUE <> PCODEXAMESANGUE AND IDDOACAO = PCODDOACAO AND IDTESTESANGUE = PCODTESTESANGUE
                                        AND RESULTADO = PRESULTADO AND PDATA = DATAEXAMESANGUE)) THEN


        BEGIN

                UPDATE EXAMESANGUE
                SET IDDOACAO = PCODDOACAO,
                IDTESTESANGUE = PCODTESTESANGUE,
                RESULTADO = PRESULTADO,
                DATAEXAMESANGUE = PDATA
                WHERE IDEXAMESANGUE = PCODEXAMESANGUE;


                SET MENSAGEM = 'Operação realizada com sucesso!';
        END;
        ELSE
                SET MENSAGEM = 'Informação de exame de sangue já existente, operação cancelada!';
```

```
        END IF;

END//
DELIMITER ;


DELIMITER //
CREATE PROCEDURE SP_INSERIRTESTESANGUE
(IN PNOME VARCHAR(150),PVALORREF DECIMAL(5,2), PTIPOSANGUINEO VARCHAR(2), PFATORRH CHAR(1),PANTICORPOS VARCHAR(150), OUT MENSAGEM VARCHAR(100))
BEGIN
        IF( (PNOME IS NULL AND PVALORREF IS NULL AND NOT EXISTS(SELECT IDTESTESANGUE
                        FROM TESTESANGUE
                        WHERE PTIPOSANGUINEO=TIPOSANGUINEO AND PFATORRH=FATORRH AND PANTICORPOS=ANTICORPOS))
                         OR
        ( PTIPOSANGUINEO IS NULL AND PFATORRH IS NULL AND PANTICORPOS IS NULL AND NOT EXISTS(SELECT IDTESTESANGUE
                        FROM TESTESANGUE
                        WHERE PNOME = NOME)) ) THEN


        BEGIN
                INSERT INTO TESTESANGUE (NOME,VALORREF,TIPOSANGUINEO,FATORRH,ANTICORPOS)
                VALUES (PNOME,PVALORREF,PTIPOSANGUINEO,PFATORRH,PANTICORPOS);

                SET MENSAGEM = 'Teste de sangue inserido com sucesso!';
        END;
        ELSE
                SET MENSAGEM = 'Ja existe um teste de sangue com esses dados e/ou Dados inseridos de forma incorreta , operação cancelada!';
        END IF;

END//
DELIMITER ;


DELIMITER //
```

```
CREATE PROCEDURE SP_ALTERATESTESANGUE
(IN PCODIGO INTEGER, PNOME VARCHAR(150), PVALORREF DECIMAL(5,2), PTIPOSANGUINEO VARCHAR(2), PFATORRH CHAR(1), PANTICORPOS VARCHAR(150), OUT MENSAGEM VARCHAR(100))
BEGIN
	IF ( ((NOT EXISTS(SELECT IDTESTESANGUE
	FROM TESTESANGUE WHERE IDTESTESANGUE <> PCODIGO AND UPPER(NOME) = UPPER(PNOME) AND PVALORREF = VALORREF )) AND PTIPOSANGUINEO IS NULL AND PFATORRH IS NULL AND PANTICORPOS IS NULL)  OR
	((NOT EXISTS(SELECT IDTESTESANGUE
	FROM TESTESANGUE WHERE IDTESTESANGUE <> PCODIGO AND
	TIPOSANGUINEO = PTIPOSANGUINEO AND FATORRH = PFATORRH AND
	ANTICORPOS = PANTICORPOS)) AND PNOME IS NULL AND PVALORREF IS NULL) ) THEN

	BEGIN
		UPDATE TESTESANGUE
			SET NOME = PNOME,
			VALORREF = PVALORREF,
			TIPOSANGUINEO = PTIPOSANGUINEO,
			FATORRH = PFATORRH,
			ANTICORPOS = PANTICORPOS
			WHERE IDTESTESANGUE = PCODIGO;


		SET MENSAGEM = 'Operação realizada com sucesso!';
	END;
	ELSE
		SET MENSAGEM = 'Informação de Teste de sangue já existente e/ou Dados inseridos de forma incorreta , operação cancelada!';
	END IF;

END//
DELIMITER ;


DELIMITER //
CREATE PROCEDURE SP_INSERIRCANDEXAME
(IN PCODIGO INTEGER,PCODIGO2 INTEGER,PRESULTADO VARCHAR(300),PDATA DATETIME, OUT MENSAGEM VARCHAR(100))
```

```
BEGIN
	IF(NOT EXISTS(SELECT IDAVALIACAO
			FROM CANDEXAME
			WHERE PCODIGO = IDEXAMES AND PCODIGO2=IDAVALIACAO)) THEN
	BEGIN
		INSERT INTO CANDEXAME
(IDEXAMES,IDAVALIACAO,RESULTADO,DATAEXAME)
		VALUES (PCODIGO,PCODIGO2,PRESULTADO,PDATA);

		SET MENSAGEM = 'Resposta do exame inserida com sucesso!';
	END;
	ELSE
		SET MENSAGEM = 'Ja existe um resultado da avaliacao cadastrado nesse exame,
operação cancelada!';
	END IF;

END//
DELIMITER ;


DELIMITER //
CREATE PROCEDURE SP_ALTERACANDEXAME
(PIDAVALIACAO INTEGER,PIDEXAMES INTEGER,PRESULTADO VARCHAR(150),PDATAEXAME
DATETIME,OUT MENSAGEM VARCHAR (150))
BEGIN
	IF(EXISTS(SELECT IDEXAMES, IDAVALIACAO
			FROM CANDEXAME
			WHERE PIDEXAMES = IDEXAMES AND PIDAVALIACAO=IDAVALIACAO)
AND
			NOT EXISTS(SELECT IDAVALIACAO
			FROM CANDEXAME
			WHERE PIDEXAMES = IDEXAMES AND PIDAVALIACAO=IDAVALIACAO
AND UPPER(PRESULTADO) = UPPER(RESULTADO))) THEN
	BEGIN
		UPDATE CANDEXAME
		SET DATAEXAME=PDATAEXAME,
		  RESULTADO=PRESULTADO
		  WHERE IDEXAMES=PIDEXAMES AND IDAVALIACAO = PIDAVALIACAO;

		SET MENSAGEM='Resposta do exame alterada!';
```

```
        END;
        ELSE
            SET MENSAGEM='Esse exame já possui esses dados, operação cancelada.';
        END IF;
END//
DELIMITER ;
```

### 2.3.4 VIEW

```
CREATE VIEW ULTIMADOACAO_TIPOSANGUINEO(DOADOR, TIPOSANGUINEO, FATORRH, ULTDOACAO)
AS
SELECT DR.NOME, T.TIPOSANGUINEO, T.FATORRH, MAX(D.DATADOACAO)
FROM DOADOR AS DR, EXAMESANGUE AS E, DOACAO AS D, TESTESANGUE AS T
WHERE TIPOSANGUINEO IS NOT NULL AND
FATORRH IS NOT NULL AND
DR.IDDOADOR = D.IDDOADOR AND
D.IDDOACAO = E.IDDOACAO AND
T.IDTESTESANGUE = E.IDTESTESANGUE
GROUP BY DR.IDDOADOR
ORDER BY T.TIPOSANGUINEO, 4;


CREATE VIEW COUNT_TIPOSANGUINEO (TIPOSANGUINEO, NUMERODOACOES)
AS
SELECT TIPOSANGUINEO, COUNT(*)
FROM TESTESANGUE AS T, DOACAO AS D, EXAMESANGUE AS E
WHERE TIPOSANGUINEO IS NOT NULL AND
CAST(DATADOACAO AS DATE) >= DATE_SUB(CURDATE(), INTERVAL 5 YEAR) AND
E.IDDOACAO = D.IDDOACAO AND T.IDTESTESANGUE = E.IDTESTESANGUE
GROUP BY TIPOSANGUINEO
ORDER BY COUNT(*);
```

# 3. CONCLUSÃO

O desenvolvimento do projeto possibilitou uma percepção de que como a modelagem de um problema a ser resolvido é sempre diferente de outros problemas de outros projetos, sendo essas diferenças, as mudanças da estrutura das tabelas e as características de cada coluna, o relacionamento entre as tabelas e se é correto deixar ON DELETE CASCADE ou ON DELETE RESTRICT nas FK's.

O grupo como todo achou um problema muito interessante, porém sentimos dificuldades para achar todas as entidades, na qual maneira seria a mais correta o relacionamento entre elas e alguns outros detalhes da colunas das tabelas, muito pelo fato de que o grupo não entende de assuntos de coletas de sangue, que por sua vez tiveram que pesquisar um pouco sobre esta área para conseguir desenvolver o projeto proposto para a turma de Banco de Dados I e como todos que são do grupo não tem experiencia no desenvolvimento de Sistemas de banco de dados, encontramos certas dificuldades que foram resolvidas mediante a pesquisa, monitoria e até mesmo de alunos com experiência na área.

Ao concluir o desenvolvimento do projeto, percebe a importância da interpretação do problema para atender a todos os requisitos e a comunicação dos integrantes, pois com o trabalho de equipe conseguimos produzir mais e melhorar o resultado final.

## 4. REFERÊNCIA

Bunn HF. Approach to the anemias In: Goldman L, Schafer AI, eds. *Goldman-Cecil Medicine*. 25th ed. Philadelphia, PA: Elsevier Saunders; 2016:chap 158.

Chernecky CC, Berger BJ. Hemoglobin (HB, Hgb). In: Chernecky CC, Berger BJ, eds. *Laboratory Tests and Diagnostic Procedures*. 6th ed. Philadelphia, PA: Elsevier; 2013:621-623.