Projeto do Banco de Dados SAM (v1.0)

Wladmir Cardoso Brandão Março 30, 2020

Resumo

O presente relatório apresenta o projeto do banco de dados SAM (Sistema Acadêmico de Matrícula). Especificamente, o relatório apresenta o resultado das atividades de especificação do minimundo, análise de requisitos, projeto conceitual, projeto lógico e projeto físico do SAM versão 1.0 (v1.0). As informações necessárias para a realização das atividades de modelagem foram coletadas a partir de especificações textuais fornecidas pelo usuário final. O banco de dados SAM (v1.0) foi concebido dentro do paradigma relacional utilizando como base o modelo relacional, sendo constituído por um conjunto de cinco entidades, cinco relacionamentos e uma média de quatro atributos por entidade. Além disso, as entidades e relacionamentos conceituais deram origem a um esquema de implementação composto por sete tabelas e sete restrições de integridade referencial entre elas. Fisicamente, o banco de dados é composto por sete arquivos indexados com índice em cada um dos campos correspondentes às chaves primárias das tabelas de origem. Adicionalmente, existem índices para cada uma das chaves estrangeiras para acelerar o processamento de transações sobre os arquivos.

1 Introdução

Instituições de ensino precisam manter seus registros acadêmicos organizados e atualizados a fim de dar suporte a alunos e professores na execução de suas atividades de aprendizagem, ensino e pesquisa. A escolha da abordagem de banco de dados utilizada para organização e manutenção desses registros acadêmicos influencia a efetividade com que as atividades são desenvolvidas. O objetivo do presente trabalho é propor um projeto de banco de dados para um sistema acadêmico de matrícula que pode ser utilizado por diversas instituições de ensino para gerenciar seus processos de matrícula de alunos em cursos. Particularmente, propõe-se uma especificação de minimundo, análise de requisitos, projeto conceitual, projeto lógico e projeto físico do banco de dados SAM (Sistema Acadêmico de Matrícula), que em sua versão 1.0 é concebido no paradigma relacional, utilizando como base o modelo relacional e podendo ser implementado em sistemas gerenciadores de banco de dados (SGBD) relacionais comerciais.

A Seção 2 apresenta a especificação de minimundo do babco de dados, incluindo uma descrição textual das principais características e restrições de dados. A Seção 3 apresenta o projeto conceitual do banco de dados, incluindo diagrama entidade-relacionamento (ER) que representa de forma gráfica as principais definições conceituais, próximas à forma como os usuários percebem o banco de dados. A Seção 4 apresenta o projeto lógico do banco de dados, incluindo diagrama relacional que representa de forma gráfica as principais definições para implementação do banco de dados dentro do paradigma relacional.

2 Especificação do Minimundo

Essa seção apresenta a descrição textual de minimundo do SAM (v1.0), um banco de dados para um sistema acadêmico que gerencia a matrícula de alunos em cursos em instituições de ensino. Em particular, os cursos da instituição de ensino são categorizados por áreas de conhecimento, sendo que um curso obrigatoriamente pertence a uma área e uma área pode possuir vários cursos. Além disso, uma área pode ser integrada por outras áreas, sendo que uma área só pode ser integrante de uma única área. As áreas são identificadas por sua sigla e possuem um nome. Os cursos são identificados por sua sigla e possuem nome, custo e professores, compostos por CPF e nome. Alunos são identificados pelo seu CPF e possuem nome, composto de primeiro nome e sobrenome, sexo e data de nascimento. Cada aluno pode se matricular em diversos cursos, sendo que em cada curso podemos ter diversos alunos matriculados, mas devemos conhecer a data e se o aluno pagou ou não a matrícula no curso. Cada curso é composto por módulos, sendo que um módulo compõe apenas um curso e não existem módulos sem vínculo a algum curso. Cada módulo é identificado por sua sigla e possui um nome. Cada módulo é composto por tópicos, sendo que cada tópico só existe em função de um módulo. Um tópico potencialmente pode ser identificado por sua sigla e possui nome e horas (carga horária do tópico). As horas do curso são derivadas da totalização das horas dos tópicos que compõem os módulos dos cursos.

2.1 Requisitos Funcionais

Diferentes grupos de usuários demandarão diferentes operações de manipulação de dados sobre diferentes porções do banco de dados. O grupo Secretaria demandará atualização e recuperação de dados sobre praticamente todas os elementos do banco de.dados, uma vez que esse grupo será o responsável por manter os dados atualizados, dando suporte aos outros grupos. O grupo Professor demandará consultas de recuperação de dados sobre suas alocações a cursos. O grupo Aluno demandará consultas para atualização de seus dados cadastrais e para manipulação de dados sobre suas matrículas. O grupo Gerência demandará a recuperação de dados agrupados sobre matrícula e cursos, tais como carga horária, custo e alocações de professores e alunos. O grupo Geral demandará recuperação de dados sobre áreas, cursos, módulos e tópicos para tomada de decisão sobre inscrição como aluno ou professor da instituição de.ensino. A Tabela 1 apresenta as principais consultas que cada grupo de usuários demandará ao sistema de banco de dados, bem como a frequência esperada de submissão (A para alta, M para média e B para baixa.

Tabela 1: Frequência esperada de consultas por grupo de usuário

Consulta		**Grupo**	**Frequência**
Q001	Visualizar os cursos em que está alocado	Professor	M
Q002	Atualizar dados cadastrais	Aluno	B
Q003	Efetivar matrícula em cursos	Aluno	B
Q004	Alterar matrícula em cursos	Aluno	B
Q005	Recuperar os cursos, e cargas horárias, em que está matriculado	Aluno	M
Q006	Visualizar lista de cursos por área de conhecimento	Geral	A
Q007	Visualizar módulos por curso	Geral	A
Q008	Visualizar tópicos por módulo do curso	Geral	A
Q009	Visualizar lista de professores por curso	Geral	M
Q010	Atualizar áreas de conhecimento	Secretaria	B
Q011	Atualizar cursos	Secretaria	A
Q012	Atualizar módulos de cursos	Secretaria	M
Q013	Atualizar tópicos de módulos de cursos	Secretaria	M
Q014	Atualizar dados cadastrais de professores	Secretaria	B
Q015	Visualizar o número total de cursos, sua carga horária e custo total por área de conhecimento	Gerência	A
Q016	Visualizar para cada curso seu custo relativo (custo absoluto por carga horária) e o número de módulos que o compõe	Gerência	B
Q017	Visualizar para cada módulo de curso, o número total de tópicos e a carga horária mínima, máxima, média e total	Gerência	B
Q018	Visualizar por área a média de carga horária e custo de cursos	Gerência	B
Q019	Visualizar cada área, seu respectivo número de nível e o número de áreas de nível imediatamente inferior	Gerência	B
Q020	Visualizar, para cada área, o curso com maior carga horária, e sua respectiva carga	Gerência	A
Q021	Visualizar, para cada área, seus cursos e suas respectivas cargas horárias e custos, para os cursos em que as cargas horárias sejam maiores que a média dos cursos da área	Gerência	A
Q022	Visualizar o número de matrículas efetuadas,	Gerência	A

3 Projeto Conceitual

Essa seção apresenta o projeto conceitual do SAM (v1.0), descrevento as principais estruturas e restrições conceituais do nanco de dados. Particularmente, a Figura 1 apresenta o diagrama entidade-relacionamento (ER) do modelo conceitual do SAM.

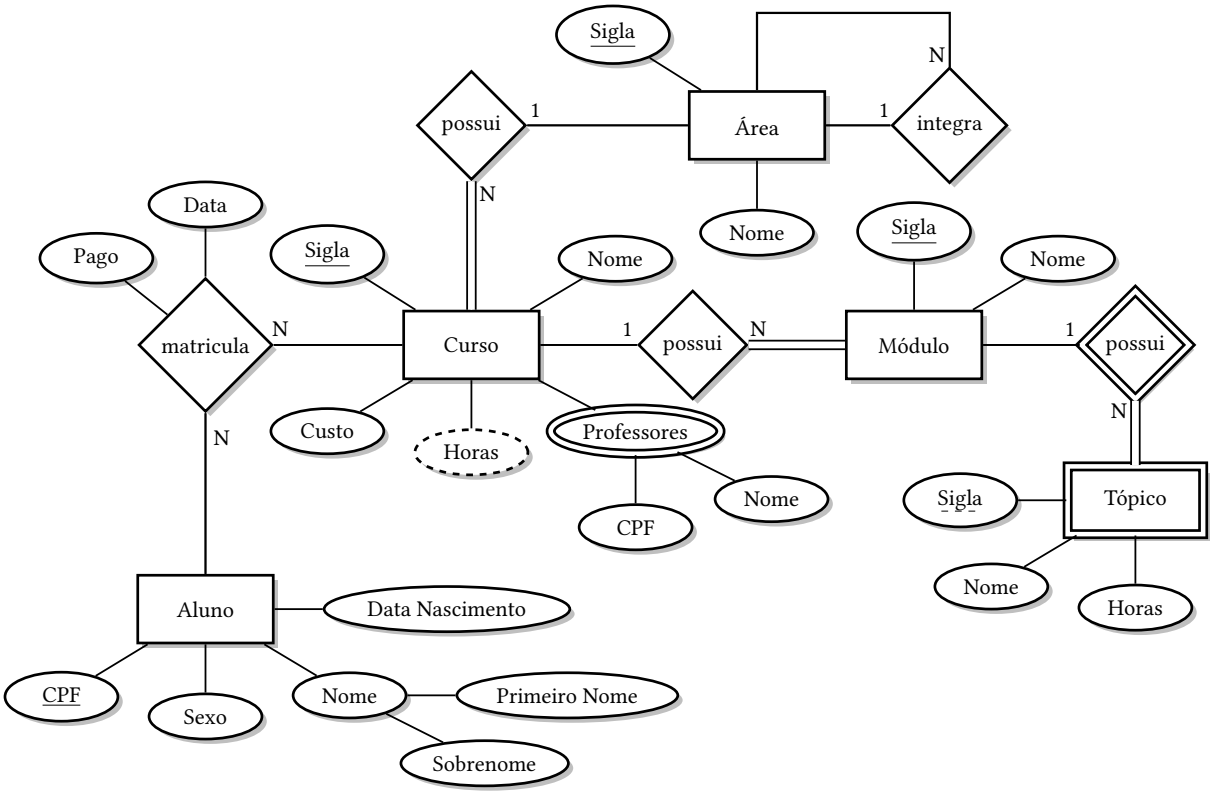

Figura 1: Diagrama ER do modelo conceitual do banco de dados SAM (v1.0)

Adicionalmente, a Tabela 2 apresenta com mais detalhes os elementos descritos no diagrama apresentados na Figura 1. Na Tabela 2, podemos observar que foram identificadas na descrição textual do minimundo cinco entidades, com uma média de quatro atributos por entidade. Além disso, foram identificados cinco relacionamentos entre entidades, bem como suas respectivas restrições de cardinalidade, e três restrições de totalidade presentes em três relacionamentos diferentes.

Tabela 2: Elementos do modelo conceitual do banco de dados SAM (v1.0)

Tipo	Subtipo	ID	Rótulo	Referência	Descrição
Entidade	Forte	E001	Aluno		
Entidade	Forte	E002	Área		
Entidade	Forte	E003	Curso		
Entidade	Forte	E004	Módulo		
Entidade	Fraca	E005	Tópico		
Relacionamento	Forte	R001	integra	E002	
Relacionamento	Forte	R002	matricula	E001, E003	
Relacionamento	Forte	R003	possui	E002, E003	
Relacionamento	Forte	R004	possui	E003, E004	
Relacionamento	Fraco	R005	possui	E004, E005	
Atributo	Chave	A001	CPF	E001	
Atributo	Composto	A002	Nome	E001	
Atributo	Simples	A003	Primeiro Nome	E001, A002	
Atributo	Simples	A004	Sobrenome	E001, A002	
Atributo	Simples	A005	Sexo	E001	
Atributo	Simples	A006	Data Nascimento	E001	
Atributo	Chave	A007	Sigla	E002	
Atributo	Simples	A008	Nome	E002	
Atributo	Chave	A009	Sigla	E003	
Atributo	Simples	A010	Nome	E003	
Atributo	Simples	A011	Custo	E003	
Atributo	Derivado	A012	Horas	E003	
Atributo	Multivalorado Composto	A013	Professores	E003	
Atributo	Simples	A014	CPF	E003, A013	
Atributo	Simples	A015	Nome	E003, A013	
Atributo	Chave	A016	Sigla	E004	
Atributo	Simples	A017	Nome	E004	
Atributo	Chave	A018	Sigla	E005	
Atributo	Simples	A019	Nome	E005	
Atributo	Simples	A020	Horas	E005	
Restrição	Totalidade	C001		R002	E003 total em R002
Restrição	Totalidade	C002		R004	E004 total em R004
Restrição	Totalidade	C003		R005	E005 total em R005
Restrição	Cardinalidade	C004	1-N	R001	E002 função *integrado* (1), E002 função *integrante* (N)
Restrição	Cardinalidade	C005	N-N	R002	E003 função *matriculante* (N), E001 função *matriculado* (N)
Restrição	Cardinalidade	C006	1-N	R003	E002 função *possuidor* (1), E003 função *possuído* (N)
Restrição	Cardinalidade	C007	1-N	R004	E003 função *possuidor* (1), E004 função *possuído* (N)
Restrição	Cardinalidade	C008	1-N	R005	E004 função *possuidor* (1), E005 função *possuído* (N)

4 Projeto Lógico

Essa seção apresenta o projeto lógico do banco de dados SAM (v1.0), descrevento as principais estruturas e restrições lógicas baseadas no modelo de implementação relacional. Particularmente, a Figura 2 apresenta o diagrama relacional do banco de dados, mapeado a partir do modelo conceitual descrito na Seção 3 do presente relatório.

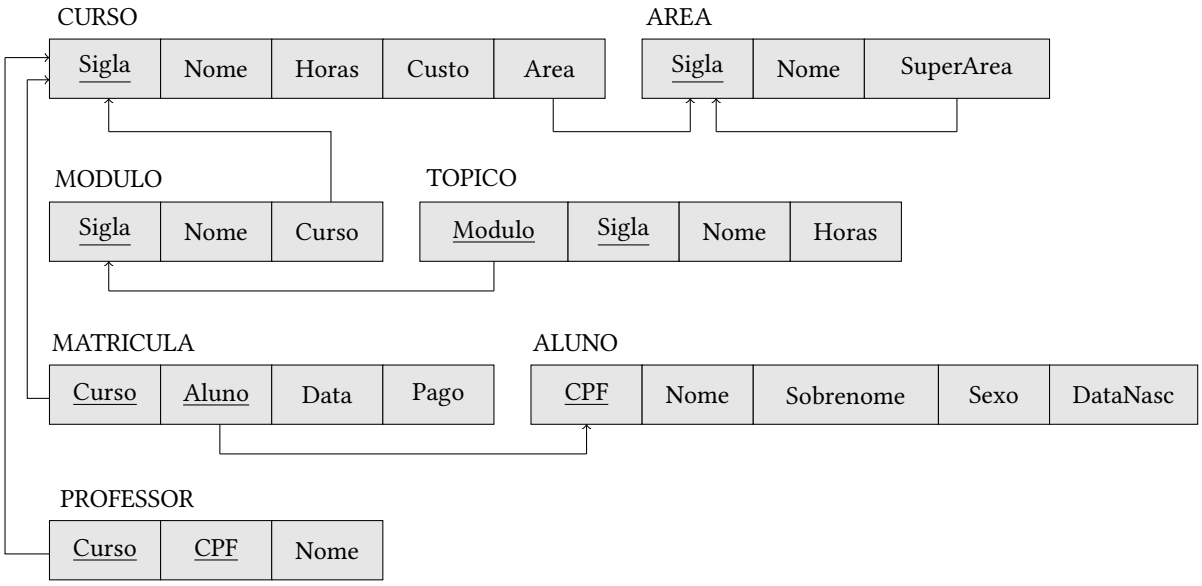

Figura 2: Diagrama do modelo de implementação relacional do SAM (v1.0)

Na Figura 2, podemos observar que foram mapeadas sete relações, com uma média de aproximadamente quatro atributos por relação. Além disso, foram identificados sete referências entre relações. O diagrama relacional apresentado na Figura 2 é muito útil para visualizar de maneira simples e compacta as relações, atributos e restrições de chave e integridade referencial presentes no banco de dados, independentemente do SGBD relacional comercial a ser adotado para sua implementação. O detalhamento de outros tipos de estruturas e restrições sobre dados são dependentes do SGBD relacional comercial a ser adotado.

Além do modelo de dados, uma importante decisão a ser tomada no projeto lógico de um banco de dados é a escolha da abordagem e da solução de banco de dados a serem adotadas. Particularmente, para implementação do SAM (v1.0) adotaremos a abordagem baseada em SGBD relacional e a solução comercial MySQL. A Figura 3 apresenta o EER do modelo de implementação relacional do SAM (v1.0), incluindo restrições de chave, representadas como uma figura amarela de chave ao lado esquerdo do rótulo do atributo, tipo, apresentada ao lado direito do rótulo do atributo, nulidade, representada como um losango ao lado esquerdo do rótulo do atributo (losango branco para NULL e azul para NOT NULL), e integridade referencial, com losango vermelho representado chaves estrangeiras.

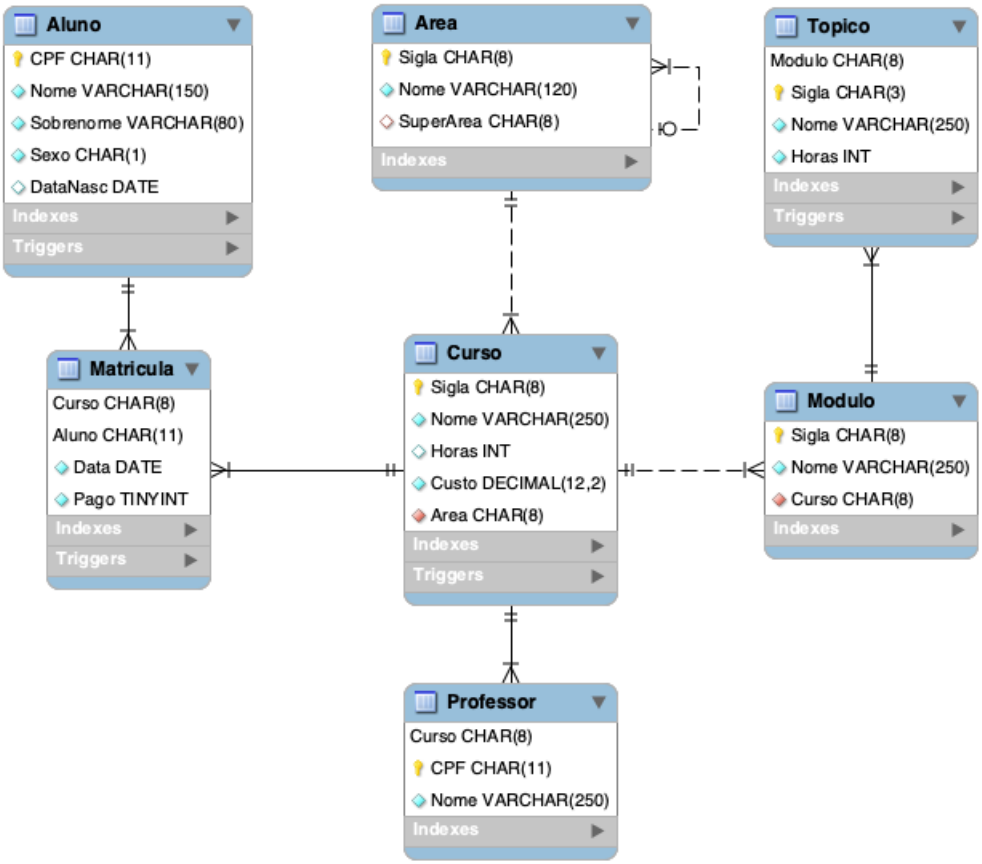

Figura 3: EER do modelo de implementação relacional do SAM (v1.0)

5 Conclusão

O presente relatório apresentou o projeto do banco de dados SAM (v1.0) para um sistema acadêmico de matrícula que, em sua versão 1.0, pode ser utilizado por diversas instituições de ensino para gerenciar seus processos de matrícula de alunos em cursos. Especificamente, propusemos uma especificação de minimundo e apresentamos os requisitos funcionais e operacionais, o projeto conceitual, lógico e físico do banco de dados, concebido no paradigma relacional e projetado para ser implementado em um SGBD relacional comercial.

6 Anexos: Scripts de Banco de Dados

***Scripts* de Criação de Tabelas**

```
-- MySQL Script generated by MySQL Workbench
-- Wed Apr  8 08:32:55 2020
-- Model: New Model    Version: 1.0
-- MySQL Workbench Forward Engineering

SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='TRADITIONAL,ALLOW_INVALID_DATES';

DROP SCHEMA IF EXISTS `SAM` ;

CREATE SCHEMA IF NOT EXISTS `SAM` DEFAULT CHARACTER SET utf8 ;

USE `SAM` ;

DROP TABLE IF EXISTS `Aluno` ;
DROP TABLE IF EXISTS `Area` ;
DROP TABLE IF EXISTS `Curso` ;
DROP TABLE IF EXISTS `Matricula` ;
DROP TABLE IF EXISTS `Modulo` ;
DROP TABLE IF EXISTS `Professor` ;
DROP TABLE IF EXISTS `Topico` ;

CREATE TABLE IF NOT EXISTS `Aluno` (
  `CPF` CHAR(11) NOT NULL,
  `Nome` VARCHAR(150) NOT NULL,
  `Sobrenome` VARCHAR(80) NOT NULL,
  `Sexo` CHAR(1) NOT NULL,
  `DataNasc` DATE)
ENGINE = InnoDB;

CREATE TABLE IF NOT EXISTS `Area` (
  `Sigla` CHAR(8) NOT NULL,
  `Nome` VARCHAR(120) NOT NULL,
  `SuperArea` CHAR(8) NULL)
ENGINE = InnoDB;

CREATE TABLE IF NOT EXISTS `Curso` (
  `Sigla` CHAR(8) NOT NULL,
  `Nome` VARCHAR(250) NOT NULL,
  `Horas` INT NULL,
  `Custo` DECIMAL(12,2) NOT NULL,
  `Area` CHAR(8) NOT NULL)
ENGINE = InnoDB;

CREATE TABLE IF NOT EXISTS `Matricula` (
  `Curso` CHAR(8) NOT NULL,
  `Aluno` CHAR(11) NOT NULL,
  `Data` DATE NOT NULL,
  `Pago` TINYINT NOT NULL DEFAULT 0)
```

```
ENGINE = InnoDB;

CREATE TABLE IF NOT EXISTS `Modulo` (
  `Sigla` CHAR(8) NOT NULL,
  `Nome` VARCHAR(250) NOT NULL,
  `Curso` CHAR(8) NOT NULL)
ENGINE = InnoDB;

CREATE TABLE IF NOT EXISTS `Professor` (
  `Curso` CHAR(8) NOT NULL,
  `CPF` CHAR(11) NOT NULL,
  `Nome` VARCHAR(250) NOT NULL)
ENGINE = InnoDB;

CREATE TABLE IF NOT EXISTS `Topico` (
  `Modulo` CHAR(8) NOT NULL,
  `Sigla` CHAR(3) NOT NULL,
  `Nome` VARCHAR(250) NOT NULL,
  `Horas` INT NOT NULL)
ENGINE = InnoDB;

SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```

***Scripts* de Criação de Restrições**

```
-- MySQL Script generated by MySQL Workbench
-- Wed Apr  8 08:32:55 2020
-- Model: New Model    Version: 1.0
-- MySQL Workbench Forward Engineering

SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='TRADITIONAL,ALLOW_INVALID_DATES';

USE `SAM`;

ALTER TABLE `Aluno` ADD CONSTRAINT `PK_Curso` PRIMARY KEY (`CPF`);
ALTER TABLE `Area` ADD CONSTRAINT `PK_Area` PRIMARY KEY (`Sigla`);
ALTER TABLE `Curso` ADD CONSTRAINT `PK_Curso` PRIMARY KEY (`Sigla`);
ALTER TABLE `Matricula` ADD CONSTRAINT `PK_Matricula` PRIMARY KEY (`Curso`, `Aluno`);
ALTER TABLE `Modulo` ADD CONSTRAINT `PK_Modulo` PRIMARY KEY (`Sigla`);
ALTER TABLE `Professor` ADD CONSTRAINT `PK_Professor` PRIMARY KEY (`Curso`, `CPF`);
ALTER TABLE `Topico` ADD CONSTRAINT `PK_Topico` PRIMARY KEY (`Modulo`,`Sigla`);

ALTER TABLE `Area` ADD CONSTRAINT `FK_Area_Area`
    FOREIGN KEY (`SuperArea`)
    REFERENCES `Area` (`Sigla`)
    ON DELETE SET NULL
    ON UPDATE CASCADE;

ALTER TABLE `Curso` ADD CONSTRAINT `FK_Curso_Area`
```

```
    FOREIGN KEY (`Area`)
    REFERENCES `Area` (`Sigla`)
    ON DELETE RESTRICT
    ON UPDATE CASCADE;

ALTER TABLE `Matricula` ADD CONSTRAINT `FK_Matricula_Curso`
    FOREIGN KEY (`Curso`)
    REFERENCES `Curso` (`Sigla`)
    ON DELETE RESTRICT
    ON UPDATE CASCADE;
ALTER TABLE `Matricula` ADD CONSTRAINT `FK_Matricula_Aluno`
    FOREIGN KEY (`Aluno`)
    REFERENCES `Aluno` (`CPF`)
    ON DELETE CASCADE
    ON UPDATE CASCADE;

ALTER TABLE `Modulo` ADD CONSTRAINT `FK_Modulo_Curso`
    FOREIGN KEY (`Curso`)
    REFERENCES `Curso` (`Sigla`)
    ON DELETE CASCADE
    ON UPDATE CASCADE;

ALTER TABLE `Professor` ADD CONSTRAINT `FK_Professor_Curso`
    FOREIGN KEY (`Curso`)
    REFERENCES `Curso` (`Sigla`)
    ON DELETE CASCADE
    ON UPDATE CASCADE;

ALTER TABLE `Topico` ADD CONSTRAINT `FK_Topico_Modulo`
    FOREIGN KEY (`Modulo`)
    REFERENCES `Modulo` (`Sigla`)
    ON DELETE CASCADE
    ON UPDATE CASCADE;

SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```

***Scripts* de Criação de Restrições (Gatilhos)**

```
-- MySQL Script generated by MySQL Workbench
-- Wed Apr  8 08:32:55 2020
-- Model: New Model    Version: 1.0
-- MySQL Workbench Forward Engineering

SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='TRADITIONAL,ALLOW_INVALID_DATES';

DELIMITER $$

USE `SAM`$$
```

```
DROP TRIGGER IF EXISTS `Aluno_BEFORE_UPDATE` $$
USE `SAM`$$
CREATE DEFINER = CURRENT_USER TRIGGER `SAM`.`Aluno_BEFORE_UPDATE` BEFORE UPDATE ON `Alun
BEGIN
  IF NEW.Sexo = 'f' THEN
SET NEW.Sexo = 'F';
  ELSEIF NEW.Sexo = 'm' THEN
SET NEW.Sexo = 'M';
  END IF;
  IF NEW.Sexo <> 'F' AND NEW.Sexo <> 'M' THEN
    SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Sexo invalido: F ou M';
  END IF;
END$$

USE `SAM`$$
DROP TRIGGER IF EXISTS `Curso_BEFORE_UPDATE` $$
USE `SAM`$$
CREATE DEFINER = CURRENT_USER TRIGGER `SAM`.`Curso_BEFORE_UPDATE` BEFORE UPDATE ON `Curs
BEGIN
  IF NEW.Horas < 1 THEN
    SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Horas invalida: Horas > 0';
  END IF;
END$$

USE `SAM`$$
DROP TRIGGER IF EXISTS `Matricula_BEFORE_UPDATE` $$
USE `SAM`$$
CREATE DEFINER = CURRENT_USER TRIGGER `SAM`.`Matricula_BEFORE_UPDATE` BEFORE UPDATE ON `
BEGIN
  IF NEW.Pago <> 0 AND NEW.Pago <> 1 THEN
    SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Pago invalido: 0 ou 1';
  END IF;
END$$

USE `SAM`$$
DROP TRIGGER IF EXISTS `Topico_BEFORE_UPDATE` $$
USE `SAM`$$
CREATE DEFINER = CURRENT_USER TRIGGER `SAM`.`Topico_BEFORE_UPDATE` BEFORE UPDATE ON `Top
BEGIN
  IF NEW.Horas < 1 THEN
    SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Horas invalida: Horas > 0';
  END IF;
END$$

DELIMITER ;

SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```

***Scripts* de Criação de Índices**

```
-- MySQL Script generated by MySQL Workbench
-- Wed Apr  8 08:32:55 2020
-- Model: New Model    Version: 1.0
-- MySQL Workbench Forward Engineering

SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='TRADITIONAL,ALLOW_INVALID_DATES';

USE `SAM`;

CREATE UNIQUE INDEX `UN_Area_Nome` ON `Area` (`Nome` ASC);
CREATE INDEX `IDX_FK_Area_Area` ON `Area` (`SuperArea` ASC);

CREATE INDEX `IDX_FK_Curso_Area` ON `Curso` (`Area` ASC);

CREATE INDEX `IDX_FK_Matricula_Aluno` ON `Matricula` (`Aluno` ASC);

CREATE INDEX `IDX_FK_Modulo_Curso` ON `Modulo` (`Curso` ASC);

CREATE INDEX `IDX_FK_Professor_Curso` ON `Professor` (`Curso` ASC);

CREATE INDEX `IDX_FK_Topico_Modulo` ON `Topico` (`Modulo` ASC);

SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```

***Scripts* de Criação do Banco de Dados**

```
USE `SAM` ;

INSERT INTO `Aluno` VALUES ('00506238866','Lisette','Friesen','F','2006-11-30'),
('00694710542','Lorine','Willms','F','2019-09-14'),
('01389490385','Orie','Daugherty','F','1986-10-25'),
('01525324976','Cali','Haag','F','1998-12-12'),
('01586644848','Jackeline','Barton','F','1994-08-16');

INSERT INTO `Area` (`Sigla`, `Nome`, `SuperArea`) VALUES ('10000003', 'Ciências Exatas e
INSERT INTO `Area` (`Sigla`, `Nome`, `SuperArea`) VALUES ('20000006', 'Ciências Biológic
INSERT INTO `Area` (`Sigla`, `Nome`, `SuperArea`) VALUES ('30000009', 'Engenharias',NULL
INSERT INTO `Area` (`Sigla`, `Nome`, `SuperArea`) VALUES ('40000001', 'Ciências da Saúde
INSERT INTO `Area` (`Sigla`, `Nome`, `SuperArea`) VALUES ('50000004', 'Ciências Agrárias
INSERT INTO `Area` (`Sigla`, `Nome`, `SuperArea`) VALUES ('60000007', 'Ciências Sociais
INSERT INTO `Area` (`Sigla`, `Nome`, `SuperArea`) VALUES ('70000000', 'Ciências Humanas'
INSERT INTO `Area` (`Sigla`, `Nome`, `SuperArea`) VALUES ('80000002', 'Lingüística, Letr
INSERT INTO `Area` (`Sigla`, `Nome`, `SuperArea`) VALUES ('90000005', 'Outra',NULL);
INSERT INTO `Area` (`Sigla`, `Nome`, `SuperArea`) VALUES ('10100008', 'Matemática','100
INSERT INTO `Area` (`Sigla`, `Nome`, `SuperArea`) VALUES ('10200002', 'Probabilidade e E
INSERT INTO `Area` (`Sigla`, `Nome`, `SuperArea`) VALUES ('10300007', 'Ciência da Comput
INSERT INTO `Area` (`Sigla`, `Nome`, `SuperArea`) VALUES ('10400001', 'Astronomia','100
```

```
INSERT INTO `Area` (`Sigla`, `Nome`, `SuperArea`) VALUES ('10500006', 'Física','1000000

INSERT INTO Curso (`Sigla`, `Nome`, `Horas`,`Custo`,`Area`) VALUES ('10101012', 'Conjunt
INSERT INTO Curso (`Sigla`, `Nome`, `Horas`,`Custo`,`Area`) VALUES ('10101020', 'Lógica
INSERT INTO Curso (`Sigla`, `Nome`, `Horas`,`Custo`,`Area`) VALUES ('10101039', 'Teoria
INSERT INTO Curso (`Sigla`, `Nome`, `Horas`,`Custo`,`Area`) VALUES ('10101047', 'Grupos
INSERT INTO Curso (`Sigla`, `Nome`, `Horas`,`Custo`,`Area`) VALUES ('10101055', 'Álgebra
INSERT INTO Curso (`Sigla`, `Nome`, `Horas`,`Custo`,`Area`) VALUES ('10101063', 'Geometr
```