UNIVERSIDADE ESTADUAL PAULISTA JÚLIO DE MESQUITA FILHO
FACULDADE DE CIÊNCIA E TECNOLOGIA
CAMPUS PRESIDENTE PRUDENTE

Projeto Semestral da Disciplina de Banco de Dados I

Sentinela - Sistema de Gerenciamento de Parque Tecnológico Empresarial

HENRIQUE FINCO FÁVERO
PAULO CELSO DOS SANTOS JÚNIOR

Presidente Prudente
2026

1. TEMA E DESCRIÇÃO DO SISTEMA

1.1 TEMA

Sentinela - Sistema de Gerenciamento de Parque Tecnológico Empresarial

1.2 DESCRIÇÃO GERAL

O Sentinela é uma plataforma web multi-tenant (projetada para atender múltiplas empresas de forma simultânea e isolada, cada uma com seus próprios dados) voltada ao gerenciamento de parques tecnológicos corporativos. Empresas se cadastram na plataforma e estruturam sua própria hierarquia organizacional, composta por filiais e departamentos. A partir dessa estrutura, cada empresa cadastra seus equipamentos de hardware (computadores, servidores, impressoras e dispositivos de rede), os componentes internos desses equipamentos e os softwares instalados em cada um, incluindo os respectivos dados de licenciamento.

O sistema mantém um catálogo global de vulnerabilidades conhecidas (CVE), associado a softwares ou a modelos de componentes de hardware. Esse catálogo é cruzado com o inventário de cada empresa para identificar quais equipamentos estão expostos a falhas de segurança já catalogadas, permitindo o monitoramento contínuo do parque tecnológico.

A plataforma também mantém o histórico de movimentação de equipamentos entre departamentos de uma mesma filial, bem como o histórico de manutenções realizadas, incluindo a substituição de componentes de hardware.

O acesso ao sistema é controlado por três perfis de usuário com escopos distintos: o dono da empresa, com acesso a todas as filiais e departamentos; o administrador de filial, com acesso restrito à sua filial; e o colaborador, com acesso restrito ao seu departamento.

1.3 OBJETIVOS DO SISTEMA

- Permitir que empresas se cadastrem na plataforma e estruturem sua hierarquia organizacional em filiais e departamentos
- Gerenciar o inventário de equipamentos de hardware e seus componentes internos

- Controlar o software instalado em cada equipamento, incluindo dados de licenciamento
- Cruzar o inventário da empresa com um catálogo de vulnerabilidades conhecidas (CVE) e identificar exposições
- Registrar o histórico de movimentação de equipamentos entre departamentos
- Registrar o histórico de manutenção e substituição de componentes de hardware
- Oferecer controle de acesso por papel, restringindo a visualização de dados conforme o escopo do usuário

1.4 PERFIS DE USUÁRIO

Dono da Empresa:

- Realiza o cadastro da empresa na plataforma
- Possui acesso irrestrito a todas as filiais, departamentos e equipamentos da empresa
- Cadastra filiais, incluindo a duplicação de uma filial existente com seus departamentos
- Cadastra usuários e define seu papel de acesso (dono, administrador de filial ou colaborador)
- Consulta relatórios e alertas de vulnerabilidade de toda a empresa

Administrador de Filial:

- Possui acesso restrito à filial à qual está vinculado
- Cadastra departamentos dentro da sua filial
- Cadastra equipamentos, componentes e softwares instalados na sua filial
- Cadastra colaboradores e concede acesso vinculado a um departamento da sua filial
- Registra manutenções, movimentações de equipamento e consulta alertas de vulnerabilidade da sua filial

Colaborador:

- Possui acesso restrito ao departamento ao qual está vinculado
- Consulta os equipamentos e softwares cadastrados no seu departamento

- Pode registrar manutenções nos equipamentos do seu departamento, conforme permissão concedida

2. REQUISITOS

2.1 REQUISITOS FUNCIONAIS

ID	Requisito
RF01	O sistema deve permitir o cadastro de uma nova empresa na plataforma
RF02	O sistema deve permitir login e logout de usuários
RF03	O sistema deve permitir que a empresa cadastre suas filiais
RF04	O sistema deve permitir duplicar uma filial existente, copiando sua estrutura de departamentos para a nova filial cadastrada. Apenas a estrutura, nunca os colaboradores
RF05	O sistema deve permitir o cadastro de departamentos dentro de uma filial
RF06	O sistema deve permitir que o dono da empresa cadastre usuários e defina seu papel (dono, administrador de filial ou colaborador)
RF07	O sistema deve permitir que um administrador de filial cadastre colaboradores vinculados a um departamento da sua filial
RF08	O sistema deve permitir o cadastro de equipamentos, vinculando-os obrigatoriamente a uma filial e a um departamento
RF09	O sistema deve permitir o cadastro de componentes de hardware vinculados a um equipamento, a partir de um catálogo de modelos de componente
RF10	O sistema deve permitir o registro de manutenção em um equipamento, incluindo a substituição de componentes

RF11	O sistema deve marcar o componente anterior como substituído ao registrar a troca por um novo, preservando o histórico
RF12	O sistema deve permitir o registro de software instalado em um equipamento, a partir de um catálogo de softwares
RF13	O sistema deve registrar dados de licenciamento (chave e validade) para cada instalação de software
RF14	O sistema deve permitir consultar o catálogo global de vulnerabilidades (CVE), filtrando por software ou modelo de componente afetado
RF15	O sistema deve identificar quais equipamentos estão expostos a uma vulnerabilidade, a partir do software instalado e dos componentes em uso
RF16	O sistema deve registrar cada ocorrência de vulnerabilidade detectada em um equipamento, com data de detecção
RF17	O sistema deve permitir atualizar o status de uma ocorrência de vulnerabilidade (pendente, corrigida ou ignorada)
RF18	O sistema deve permitir reabertura de uma ocorrência de vulnerabilidade já corrigida, gerando um novo registro de detecção
RF19	O sistema deve permitir o registro de movimentação de um equipamento entre departamentos de uma mesma filial
RF20	O sistema deve exibir o histórico de movimentações de um equipamento
RF21	O sistema deve exibir o histórico de manutenções de um equipamento, incluindo os componentes trocados em cada uma
RF22	O sistema deve restringir a visualização de dados de acordo com o escopo do usuário logado (empresa, filial ou departamento)

OBS: Não especificamos aqui os requisitos de Update e Delete para as entidades que serão cadastradas (usuário, filial, departamento, equipamento) pois ia ficar uma lista de requisitos gigante e creio que não é o foco aqui e sim o Banco de Dados.

2.2 REQUISITOS NÃO FUNCIONAIS

ID	Requisito
RNF01	As senhas dos usuários devem ser armazenadas utilizando algoritmo de hash seguro (BCrypt)
RNF02	A conexão com o banco de dados MySQL deve ser feita via conector nativo, sem uso de ORM ou frameworks de persistência
RNF03	Todas as queries ao banco de dados devem ser escritas explicitamente em SQL, utilizando PreparedStatements para prevenção de SQL Injection
RNF04	O backend deve ser desenvolvido em Python + FastAPI, expondo uma API REST consumida pelo frontend
RNF05	A interface do usuário deve ser funcional, responsiva e intuitiva, desenvolvida em HTML, CSS e JavaScript puro
RNF06	O banco de dados deve utilizar MySQL como SGBD
RNF07	O sistema deve impedir o acesso de um usuário a dados de uma empresa diferente da sua
RNF08	O sistema deve impedir que um administrador de filial visualize ou edite dados de outra filial da mesma empresa
RNF09	A movimentação de equipamento deve ser restrita a departamentos pertencentes à mesma filial
RNF10	O catálogo de software, modelos de componente e vulnerabilidades deve ser compartilhado entre todas as empresas cadastradas na plataforma
RNF11	Toda vulnerabilidade cadastrada deve referenciar exatamente um software ou um modelo de componente, nunca os dois simultaneamente

RNF12	A substituição de um componente deve preservar o registro do componente anterior, nunca removendo-o do histórico

3. MODELAGEM DO BANCO DE DADOS

3.1 VISÃO GERAL DAS ENTIDADES

Para facilitar a visualização sobre as entidades do banco de dados podemos dizer que temos quatro grandes grupos de entidades representadas aqui:

Grupo Organizacional contendo a estrutura hierárquica de cada empresa cadastrada na plataforma: EMPRESA, FILIAL, DEPARTAMENTO, USUARIO.

Grupo de Hardware contemplando o inventário físico e seus componentes internos: EQUIPAMENTO, MODELO_COMPONENTE, COMPONENTE.

Grupo de Software que fica responsável pelo catálogo de softwares e instalações em equipamentos: SOFTWARE, EQUIPAMENTO_SOFTWARE.

Por última o grupo de Segurança e Operação contendo as entidades que representam vulnerabilidades e históricos de eventos sobre os equipamentos: VULNERABILIDADE, OCORRENCIA_VULNERABILIDADE, MANUTENCAO, MOVIMENTACAO_EQUIPAMENTO.

3.2 DESCRIÇÃO DAS ENTIDADES

Entidade	Atributos	Descrição
EMPRESA	{id, razao_social, cnpj, email, criado_em}	Representa cada empresa cadastrada na plataforma
FILIAL	{id, empresa_id, nome, cidade, uf, endereco}	Representa uma unidade física de uma empresa
DEPARTAMENTO	{id, filial_id, nome}	Representa um setor dentro de uma filial

USUARIO	{id, empresa_id, filial_id, departamento_id, nome, email, senha, tipo_acesso, criado_em}	Representa os usuários com acesso ao sistema. O escopo de acesso é definido pelo tipo_acesso e pelas FKs de filial/departamento preenchidas
EQUIPAMENTO	{id, filial_id, departamento_id, tipo, patrimonio, hostname, endereco_ip, status, criado_em}	Representa um ativo de hardware (computador, servidor, impressora ou equipamento de rede)
MODELO_COMPONENTE	{id, tipo, fabricante, modelo}	Catálogo global de modelos de componentes de hardware (CPU, RAM, HD etc.), compartilhado entre empresas
COMPONENTE	{id, equipamento_id, modelo_componente_id, manutencao_id, status, data_instalacao, data_remocao}	Representa uma peça física instalada em um equipamento. Mantém histórico: ao ser substituída, o registro é marcado como substituído e um novo é criado
SOFTWARE	{id, nome, fabricante, versao}	Catálogo global de softwares e suas

		versões, compartilhado entre empresas
EQUIPAMENTO_SOFTWARE	{id, equipamento_id, software_id, data_instalacao, chave_licenca, validade_licenca}	Representa uma instalação específica de um software em um equipamento, com dados de licenciamento por instalação
VULNERABILIDADE	{id, software_id, componente_modelo_id, codigo_cve, severidade, descricao}	Catálogo global de vulnerabilidades conhecidas (CVE). Cada registro refere-se exclusivamente a um software ou a um modelo de componente, nunca a ambos
OCORRENCIA_VULNERABILIDADE	{id, equipamento_id, vulnerabilidade_id, data_deteccao, status, data_resolucao}	Registra cada detecção de uma vulnerabilidade em um equipamento específico. Cada nova detecção gera um novo registro, permitindo reabertura

MANUTENCAO	{id, equipamento_id, data, descricao, status}	Registra um evento de manutenção realizado em um equipamento
MOVIMENTACAO_EQUIPAMENTO	{id, equipamento_id, departamento_origem_id, departamento_destino_id, data, motivo}	Registra a movimentação de um equipamento entre departamentos de uma mesma filial

3.3 MAPEAMENTO DOS RELACIONAMENTOS

Relacionamento	Card.	Descrição
EMPRESA → FILIAL	(1,1) → (0,n)	Cada empresa possui várias filiais. Cada filial pertence a uma empresa.
EMPRESA → USUARIO	(1,1) → (0,n)	Cada empresa possui vários usuários. Cada usuário pertence a uma empresa.
FILIAL → DEPARTAMENTO	(1,1) → (0,n)	Cada filial possui vários departamentos. Cada departamento pertence a uma filial.
FILIAL → USUARIO (admin)	(0,1) → (0,n)	Uma filial pode ter vários administradores. Um usuário admin de filial pertence a uma filial (nulo para dono da empresa).
DEPARTAMENTO → USUARIO (colaborador)	(0,1) → (0,n)	Um departamento pode ter vários colaboradores. Um colaborador pertence a um departamento (nulo para os demais papéis).

FILIAL → EQUIPAMENTO	(1,1) → (0,n)	Cada filial possui vários equipamentos. Cada equipamento pertence obrigatoriamente a uma filial.
DEPARTAMENTO → EQUIPAMENTO	(0,1) → (0,n)	Um departamento pode ter vários equipamentos. Um equipamento pode, ou não, estar vinculado a um departamento.
EQUIPAMENTO → COMPONENTE	(1,1) → (0,n)	Cada equipamento possui vários componentes ao longo do tempo. Cada componente pertence a um equipamento.
MODELO_COMPONENTE → COMPONENTE	(1,1) → (0,n)	Um modelo de componente pode estar instalado em várias peças físicas. Cada componente refere-se a um modelo.
MANUTENCAO → COMPONENTE	(0,1) → (0,n)	Uma manutenção pode ter originado a troca de vários componentes. Um componente pode, ou não, ter sido instalado em uma manutenção.
EQUIPAMENTO → MANUTENCAO	(1,1) → (0,n)	Cada equipamento possui vários registros de manutenção. Cada manutenção pertence a um equipamento.
EQUIPAMENTO → EQUIPAMENTO_SOFTWARE	(1,1) → (0,n)	Cada equipamento possui várias instalações de software. Cada instalação pertence a um equipamento.

SOFTWARE → EQUIPAMENTO_SOFTWARE	(1,1) → (0,n)	Um software pode estar instalado em vários equipamentos. Cada instalação refere-se a um software.
SOFTWARE → VULNERABILIDADE	(0,1) → (0,n)	Um software pode ter várias vulnerabilidades conhecidas. Cada vulnerabilidade de software refere-se a um software (nulo se for vulnerabilidade de hardware).
MODELO_COMPONENTE → VULNERABILIDADE	(0,1) → (0,n)	Um modelo de componente pode ter várias vulnerabilidades conhecidas. Cada vulnerabilidade de hardware refere-se a um modelo (nulo se for vulnerabilidade de software).
EQUIPAMENTO → OCORRENCIA_VULNERABILIDADE	(1,1) → (0,n)	Cada equipamento pode ter várias ocorrências de vulnerabilidade detectadas. Cada ocorrência refere-se a um equipamento.
VULNERABILIDADE → OCORRENCIA_VULNERABILIDADE	(1,1) → (0,n)	Uma vulnerabilidade pode ter sido detectada em várias ocorrências. Cada ocorrência refere-se a uma vulnerabilidade.
EQUIPAMENTO → MOVIMENTACAO_EQUIPAMENTO	(1,1) → (0,n)	Cada equipamento pode ter várias movimentações registradas. Cada movimentação refere-se a um equipamento.
DEPARTAMENTO → MOVIMENTACAO_EQUIPAMENTO (origem)	(1,1) → (0,n)	Um departamento pode ser origem de várias movimentações. Cada movimentação tem um departamento de origem.

DEPARTAMENTO → MOVIMENTACAO_EQUIPAMENTO (destino)	(1,1) → (0,n)	Um departamento pode ser destino de várias movimentações. Cada movimentação tem um departamento de destino.

3.4 DIAGRAMAS

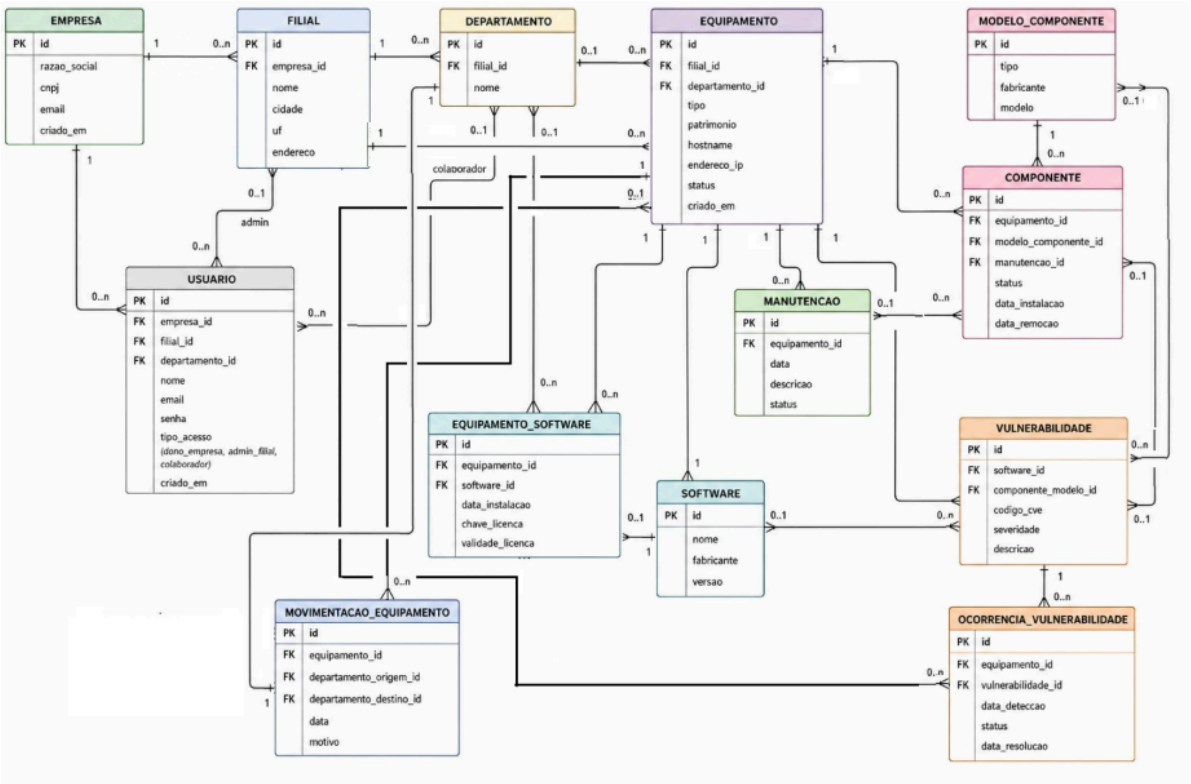

Diagrama Entidade-Relacionamento Criado pelos Autores do Projeto - Acervo Pessoal
Para melhor visualização, consulte o link: Visualizar Diagrama

3.5 MAPEAMENTO PARA MODELO RELACIONAL

O mapeamento do DER para o modelo relacional é o processo de transformar o modelo conceitual em tabelas, colunas e chaves prontas para implementação. Cada entidade vira uma tabela, cada atributo vira uma coluna, e o atributo identificador vira a chave primária.

Para os relacionamentos, a cardinalidade determina como a conversão é feita. Nos relacionamentos 1:1, a chave primária de uma das entidades migra como chave estrangeira para a tabela da outra. Nos relacionamentos 1:N, a chave

primária do lado “um” migra como chave estrangeira para a tabela do lado “muitos”. Já nos relacionamentos N:M, não é possível representar a associação diretamente nas duas tabelas existentes, então uma terceira tabela associativa é criada contendo as chaves estrangeiras de ambas as entidades participantes.

Uma observação sobre o projeto: a entidade VULNERABILIDADE possui duas chaves estrangeiras opcionais (software_id e componente_modelo_id), das quais exatamente uma deve estar preenchida. Essa exclusividade é garantida por uma CHECK constraint no banco de dados, detalhada na seção 4.

3.5.1 FORMATO TEXTUAL CLÁSSICO

EMPRESA = {id, razao_social, cnpj, email, criado_em}
FILIAL = {id, empresa_id*, nome, cidade, uf, endereco}
DEPARTAMENTO = {id, filial_id*, nome}
USUARIO = {id, empresa_id*, filial_id*, departamento_id*, nome, email, senha, tipo_acesso, criado_em}
EQUIPAMENTO = {id, filial_id*, departamento_id*, tipo, patrimonio, hostname, endereco_ip, status, criado_em}
MODELO_COMPONENTE = {id, tipo, fabricante, modelo}
COMPONENTE = {id, equipamento_id*, modelo_componente_id*, manutencao_id*, status, data_instalacao, data_remocao}
SOFTWARE = {id, nome, fabricante, versao}
EQUIPAMENTO_SOFTWARE = {id, equipamento_id*, software_id*, data_instalacao, chave_licenca, validade_licenca}
VULNERABILIDADE = {id, software_id*, componente_modelo_id*, codigo_cve, severidade, descricao}
OCORRENCIA_VULNERABILIDADE = {id, equipamento_id*, vulnerabilidade_id*, data_deteccao, status, data_resolucao}

MANUTENCAO = {id, equipamento_id*, data, descricao, status}
MOVIMENTACAO_EQUIPAMENTO = {id, equipamento_id*, departamento_origem_id*, departamento_destino_id*, data, motivo}

3.5.2 FORMATO DE TABELAS

EMPRESA
INT id (PK)
VARCHAR(255) razao_social
VARCHAR(18) cnpj
VARCHAR(255) email
DATETIME criado_em

FILIAL
INT id (PK)
INT empresa_id (FK -> EMPRESA)
VARCHAR(255) nome
VARCHAR(100) cidade
CHAR(2) uf
VARCHAR(255) endereco

DEPARTAMENTO
INT id (PK)
INT filial_id (FK -> FILIAL)
VARCHAR(255) nome

USUARIO
INT id (PK)

INT empresa_id (FK -> EMPRESA)
INT filial_id (FK -> FILIAL, nullable)
INT departamento_id (FK -> DEPARTAMENTO, nullable)
VARCHAR(255) nome
VARCHAR(255) email
VARCHAR(255) senha
ENUM tipo_acesso ('dono_empresa', 'admin_filial', 'colaborador')
DATETIME criado_em

EQUIPAMENTO
INT id (PK)
INT filial_id (FK -> FILIAL)
INT departamento_id (FK -> DEPARTAMENTO, nullable)
ENUM tipo ('desktop', 'notebook', 'servidor', 'impressora', 'switch', 'roteador', 'outro')
VARCHAR(100) patrimonio
VARCHAR(100) hostname
VARCHAR(45) endereco_ip
ENUM status ('ativo', 'manutencao', 'inativo')
DATETIME criado_em

MODELO_COMPONENTE
INT id (PK)
ENUM tipo ('cpu', 'ram', 'hd', 'ssd', 'placa_mae', 'fonte', 'outro')
VARCHAR(255) fabricante
VARCHAR(255) modelo

COMPONENTE
INT id (PK)
INT equipamento_id (FK -> EQUIPAMENTO)

INT modelo_componente_id (FK -> MODELO_COMPONENTE)
INT manutencao_id (FK -> MANUTENCAO, nullable)
ENUM status ('ativo', 'substituido')
DATETIME data_instalacao
DATETIME data_remocao (nullable)

SOFTWARE
INT id (PK)
VARCHAR(255) nome
VARCHAR(255) fabricante
VARCHAR(50) versao

EQUIPAMENTO_SOFTWARE
INT id (PK)
INT equipamento_id (FK -> EQUIPAMENTO)
INT software_id (FK -> SOFTWARE)
DATETIME data_instalacao
VARCHAR(255) chave_licenca
DATE validade_licenca

VULNERABILIDADE
INT id (PK)
INT software_id (FK -> SOFTWARE, nullable)
INT componente_modelo_id (FK -> MODELO_COMPONENTE, nullable)
VARCHAR(20) codigo_cve
ENUM severidade ('baixa', 'media', 'alta', 'critica')
TEXT descricao
CHECK: exatamente um entre software_id e componente_modelo_id deve ser não nulo

OCORRENCIA_VULNERABILIDADE
INT id (PK)
INT equipamento_id (FK -> EQUIPAMENTO)
INT vulnerabilidade_id (FK -> VULNERABILIDADE)
DATETIME data_deteccao
ENUM status ('pendente', 'corrigido', 'ignorado')
DATETIME data_resolucao (nullable)

MANUTENCAO
INT id (PK)
INT equipamento_id (FK -> EQUIPAMENTO)
DATETIME data
TEXT descricao
ENUM status ('aberta', 'concluida')

MOVIMENTACAO_EQUIPAMENTO
INT id (PK)
INT equipamento_id (FK -> EQUIPAMENTO)
INT departamento_origem_id (FK -> DEPARTAMENTO)
INT departamento_destino_id (FK -> DEPARTAMENTO)
DATETIME data
VARCHAR(255) motivo

4. IMPLEMENTAÇÃO DO BANCO DE DADOS

4.1 SCRIPT SQL PARA IMPLEMENTAÇÃO

No script abaixo as chaves estrangeiras foram definidas com políticas explícitas de ON DELETE e ON UPDATE. Para entidades que compõem o histórico do sistema, adotou-se ON DELETE RESTRICT: uma filial ou departamento não pode ser excluído enquanto ainda possuir equipamentos vinculados e um

equipamento não pode ser excluído enquanto possuir componentes, manutenções, instalações de software ou ocorrência de vulnerabilidades associadas.

Para vínculos opcionais por definição do modelo (como o departamento_id de usuário e equipamento), adotou-se ON DELETE SET NULL, permitindo a remoção do registro pai sem comprometer o registro dependente. Em todos os casos, ON UPDATE CASCADE garante a propagação automática caso uma chave primária seja alterada.

```SQL
CREATE DATABASE sentinela;

USE sentinela;

CREATE TABLE empresa (
    id INT AUTO_INCREMENT PRIMARY KEY,
    razao_social VARCHAR(255) NOT NULL,
    cnpj VARCHAR(18) NOT NULL UNIQUE,
    email VARCHAR(255) NOT NULL UNIQUE,
    criado_em DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP
);

CREATE TABLE filial (
    id INT AUTO_INCREMENT PRIMARY KEY,
    empresa_id INT NOT NULL,
    nome VARCHAR(255) NOT NULL,
    cidade VARCHAR(100) NOT NULL,
    uf CHAR(2) NOT NULL,
    endereco VARCHAR(255) NOT NULL,
    FOREIGN KEY (empresa_id) REFERENCES empresa(id)
        ON DELETE RESTRICT
        ON UPDATE CASCADE
);

CREATE TABLE departamento (
    id INT AUTO_INCREMENT PRIMARY KEY,
    filial_id INT NOT NULL,
    nome VARCHAR(255) NOT NULL,
    FOREIGN KEY (filial_id) REFERENCES filial(id)
        ON DELETE RESTRICT
        ON UPDATE CASCADE
);

CREATE TABLE usuario (
    id INT AUTO_INCREMENT PRIMARY KEY,
    empresa_id INT NOT NULL,
    filial_id INT NULL,
    departamento_id INT NULL,
```

```
    nome VARCHAR(255) NOT NULL,
    email VARCHAR(255) NOT NULL UNIQUE,
    senha VARCHAR(255) NOT NULL,
    tipo_acesso ENUM('dono_empresa', 'admin_filial', 'colaborador') NOT NULL,
    criado_em DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (empresa_id) REFERENCES empresa(id)
        ON DELETE RESTRICT
        ON UPDATE CASCADE,
    FOREIGN KEY (filial_id) REFERENCES filial(id)
        ON DELETE SET NULL
        ON UPDATE CASCADE,
    FOREIGN KEY (departamento_id) REFERENCES departamento(id)
        ON DELETE SET NULL
        ON UPDATE CASCADE
);

CREATE TABLE equipamento (
    id INT AUTO_INCREMENT PRIMARY KEY,
    filial_id INT NOT NULL,
    departamento_id INT NULL,
    tipo ENUM('desktop', 'notebook', 'servidor', 'impressora', 'switch',
'roteador', 'outro') NOT NULL,
    patrimonio VARCHAR(100) NOT NULL UNIQUE,
    hostname VARCHAR(100) NULL,
    endereco_ip VARCHAR(45) NULL,
    status ENUM('ativo', 'manutencao', 'inativo') NOT NULL DEFAULT 'ativo',
    criado_em DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (filial_id) REFERENCES filial(id)
        ON DELETE RESTRICT
        ON UPDATE CASCADE,
    FOREIGN KEY (departamento_id) REFERENCES departamento(id)
        ON DELETE SET NULL
        ON UPDATE CASCADE
);

CREATE TABLE modelo_componente (
    id INT AUTO_INCREMENT PRIMARY KEY,
    tipo ENUM('cpu', 'ram', 'hd', 'ssd', 'placa_mae', 'fonte', 'outro') NOT
NULL,
    fabricante VARCHAR(255) NOT NULL,
    modelo VARCHAR(255) NOT NULL
);

CREATE TABLE manutencao (
    id INT AUTO_INCREMENT PRIMARY KEY,
    equipamento_id INT NOT NULL,
    data DATETIME NOT NULL,
    descricao TEXT NOT NULL,
```

```
    status ENUM('aberta', 'concluida') NOT NULL DEFAULT 'aberta',
    FOREIGN KEY (equipamento_id) REFERENCES equipamento(id)
        ON DELETE RESTRICT
        ON UPDATE CASCADE
);

CREATE TABLE componente (
    id INT AUTO_INCREMENT PRIMARY KEY,
    equipamento_id INT NOT NULL,
    modelo_componente_id INT NOT NULL,
    manutencao_id INT NULL,
    status ENUM('ativo', 'substituido') NOT NULL DEFAULT 'ativo',
    data_instalacao DATETIME NOT NULL,
    data_remocao DATETIME NULL,
    FOREIGN KEY (equipamento_id) REFERENCES equipamento(id)
        ON DELETE RESTRICT
        ON UPDATE CASCADE,
    FOREIGN KEY (modelo_componente_id) REFERENCES modelo_componente(id)
        ON DELETE RESTRICT
        ON UPDATE CASCADE,
    FOREIGN KEY (manutencao_id) REFERENCES manutencao(id)
        ON DELETE SET NULL
        ON UPDATE CASCADE
);

CREATE TABLE software (
    id INT AUTO_INCREMENT PRIMARY KEY,
    nome VARCHAR(255) NOT NULL,
    fabricante VARCHAR(255) NOT NULL,
    versao VARCHAR(50) NOT NULL
);

CREATE TABLE equipamento_software (
    id INT AUTO_INCREMENT PRIMARY KEY,
    equipamento_id INT NOT NULL,
    software_id INT NOT NULL,
    data_instalacao DATETIME NOT NULL,
    chave_licenca VARCHAR(255) NULL,
    validade_licenca DATE NULL,
    FOREIGN KEY (equipamento_id) REFERENCES equipamento(id)
        ON DELETE RESTRICT
        ON UPDATE CASCADE,
    FOREIGN KEY (software_id) REFERENCES software(id)
        ON DELETE RESTRICT
        ON UPDATE CASCADE
);

CREATE TABLE vulnerabilidade (
```

```
    id INT AUTO_INCREMENT PRIMARY KEY,
    software_id INT NULL,
    componente_modelo_id INT NULL,
    codigo_cve VARCHAR(20) NOT NULL UNIQUE,
    severidade ENUM('baixa', 'media', 'alta', 'critica') NOT NULL,
    descricao TEXT NOT NULL,
    FOREIGN KEY (software_id) REFERENCES software(id)
        ON DELETE RESTRICT ON UPDATE RESTRICT,
    FOREIGN KEY (componente_modelo_id) REFERENCES modelo_componente(id)
        ON DELETE RESTRICT ON UPDATE RESTRICT,
    CHECK (
        (software_id IS NOT NULL AND componente_modelo_id IS NULL)
        OR
        (software_id IS NULL AND componente_modelo_id IS NOT NULL)
    )
);

CREATE TABLE ocorrencia_vulnerabilidade (
    id INT AUTO_INCREMENT PRIMARY KEY,
    equipamento_id INT NOT NULL,
    vulnerabilidade_id INT NOT NULL,
    data_deteccao DATETIME NOT NULL,
    status ENUM('pendente', 'corrigido', 'ignorado') NOT NULL DEFAULT
'pendente',
    data_resolucao DATETIME NULL,
    FOREIGN KEY (equipamento_id) REFERENCES equipamento(id)
        ON DELETE RESTRICT ON UPDATE CASCADE,
    FOREIGN KEY (vulnerabilidade_id) REFERENCES vulnerabilidade(id)
        ON DELETE RESTRICT ON UPDATE CASCADE
);

CREATE TABLE movimentacao_equipamento (
    id INT AUTO_INCREMENT PRIMARY KEY,
    equipamento_id INT NOT NULL,
    departamento_origem_id INT NOT NULL,
    departamento_destino_id INT NOT NULL,
    data DATETIME NOT NULL,
    motivo VARCHAR(255) NULL,
    FOREIGN KEY (equipamento_id) REFERENCES equipamento(id)
        ON DELETE RESTRICT
        ON UPDATE CASCADE,
    FOREIGN KEY (departamento_origem_id) REFERENCES departamento(id)
        ON DELETE RESTRICT
        ON UPDATE CASCADE,
    FOREIGN KEY (departamento_destino_id) REFERENCES departamento(id)
        ON DELETE RESTRICT
        ON UPDATE CASCADE
);
```