

Proteção de Pessoas Negligenciadas

Bases de Dados - EIC0023

Ano letivo 2020-21



*Tomás
Vicente
up201904609*

*Gabriel
Martins
up201906072*

*António
Ribeiro
up201906761*



Índice

Descrição do Tema	2
Elementos Principais	3
Funcionamento da organização	6
Apoios	10
Contribuições	13
UML	14
UML Revisto	16
Modelo Relacional	18
Dependências Funcionais	33
Formas Normais	35



Descrição do Tema

Pretende-se modelar a base de dados que permitirá armazenar os dados necessários ao funcionamento de uma organização não governamental - PPN (Proteção de Pessoas Negligenciadas). Para isso, será necessário representar todos os domínios da atividade, bem como os recursos humanos e materiais disponíveis.



Elementos Principais

Identificação Geográfica

Para otimizar a prestação de apoios, é necessário ter um bom conhecimento da distribuição geográfica da rede de necessitados e colaboradores. Assim, é essencial registar a morada de todos os intervenientes, bem como a sua localidade e país. A cada localidade corresponde um código de zona que agiliza todo o processo.

Pessoas

Sobre todas as pessoas envolvidas nestas atividades, é relevante guardar uma série de atributos identificativos como, por exemplo:

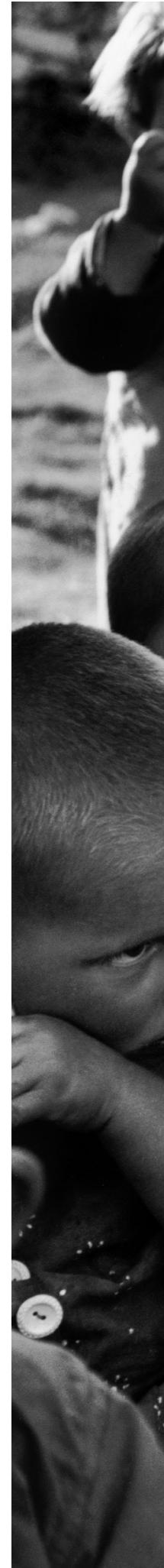
- Nome
- Data de nascimento
- NIF
- Morada
- Número de telefone

Prevendo que alguns destes atributos sejam difíceis de recolher, considera-se de extrema importância recolher o nome e data de nascimento.

Necessitados

Todas as pessoas que beneficiem de algum tipo de apoio por parte da instituição são objeto de registo para efeitos de organização e alocação de recursos.

Além dos atributos previamente mencionados, é necessário contabilizar o rendimento atual do cidadão, sendo este um dos principais critérios de decisão nos pedidos de apoio.





Funcionamento da organização

Recursos Humanos

Com a óbvia exceção dos voluntários, todos os colaboradores têm associado um horário de trabalho que deve obedecer às normas legais, não ultrapassando as 8h diárias.

Voluntários

Os voluntários são uma parte integrante do funcionamento da organização. Quando uma atividade exige um grande número de pessoas ou a rede de colaboradores numa área é insuficiente, a organização recorre a voluntários. Estas pessoas colaboram, frequentemente, no auxílio aos restantes trabalhadores, nomeadamente na organização e distribuição dos produtos nos armazéns e no auxílio aos trabalhadores dos abrigos (poderá apenas trabalhar num estabelecimento de cada tipo).





Orientadores

Os orientadores são os verdadeiros responsáveis por garantir que os apoios chegam a quem deles necessita. Todos os apoios são coordenados por um orientador que assegura a sua execução em tempo útil.

Por forma a garantir o bom funcionamento da organização, os orientadores oferecem, regularmente, formações aos voluntários, sendo cada formação limitada a 10 voluntários para garantir um acompanhamento personalizado.

Administradores

Para uma gestão eficiente dos recursos da organização, estão presentes administradores a quem compete todas as questões burocráticas e a gestão dos armazéns. Estes funcionários são também responsáveis por aprovar todos os pedidos de apoio com base em critérios predefinidos.

Recursos Materiais

Armazéns

O armazenamento dos produtos está, naturalmente, condicionado ao espaço disponível. Os armazéns da organização têm uma capacidade própria, estando o armazenamento de produtos dependente da sua quantidade e tamanho, bem como da capacidade disponível em armazém. Quando de uma contribuição ou apoio, esta capacidade é atualizada, sendo que o espaço ocupado é igual a número_produtos * dimensão_produto.

Para facilitar as atividades de distribuição e armazenamento, o armazém está dividido por secções para os diferentes tipos de produtos, posteriormente organizados pelas suas categorias (e.g. por conteúdo e data de validade no caso de produtos alimentares, por tamanho no caso de vestuário e por género no caso dos produtos de higiene).

Por serem descentralizados, importa também ter conhecimento da morada de cada um dos armazéns.

Abrigos

No sentido de assegurar apoios a grupos vulneráveis, a organização dispõe de um conjunto de abrigos dispersos por várias localidades e equipados com as condições adequadas. Estes abrigos contêm um número limitado de camas, sendo necessária uma gestão cuidada dos seus ocupantes.



Apoios

Tipos de apoios

Uma vez que as necessidades da população são extremamente diversificadas, é necessário que os apoios também o sejam. Por isso, a organização oferece uma variedade de apoios, nomeadamente:

- Produtos alimentares de vários tipos:
 - enlatados;
 - bebidas;
 - laticínios;
 - integrais.
- Produtos de higiene pessoal;
- Produtos de vestuário de vários tamanhos;
- Alojamento.

Os apoios são concedidos por um determinado período de tempo, contabilizando-se o início e o fim do mesmo.



Por vezes é também fornecido um apoio financeiro (subsídio) a todos aqueles que se encontram em situação precária, para que possam realizar pequenos pagamentos como os da luz ou gás, conseguindo garantir condições básicas para permanecer nas suas residências. Estes apoios dependem do saldo bancário que a instituição tem disponível que resulta do balanço entre o total da recolha de doações monetárias e os apoios já fornecidos.



Pedidos de apoio

Assegurando uma distribuição justa dos recursos escassos, a atribuição de apoios está sujeita a condições restritas. Por essa razão, é necessária a submissão de um pedido de apoio por parte de cada necessitado. Nesse pedido, devem constar as razões que o fundamentam bem como o tipo de necessidade.

Aquando da receção do pedido, existe uma avaliação por parte de um administrador, sendo-lhe atribuída uma prioridade na escala [0, 10]. Esta prioridade ditará se o apoio será concedido ou entrará em lista de espera para posterior análise.

A cada necessitado apenas podem corresponder, no máximo, 5 pedidos de apoio em simultâneo, de forma a não sobrecarregar a organização com um excesso de pedidos.



Contribuições



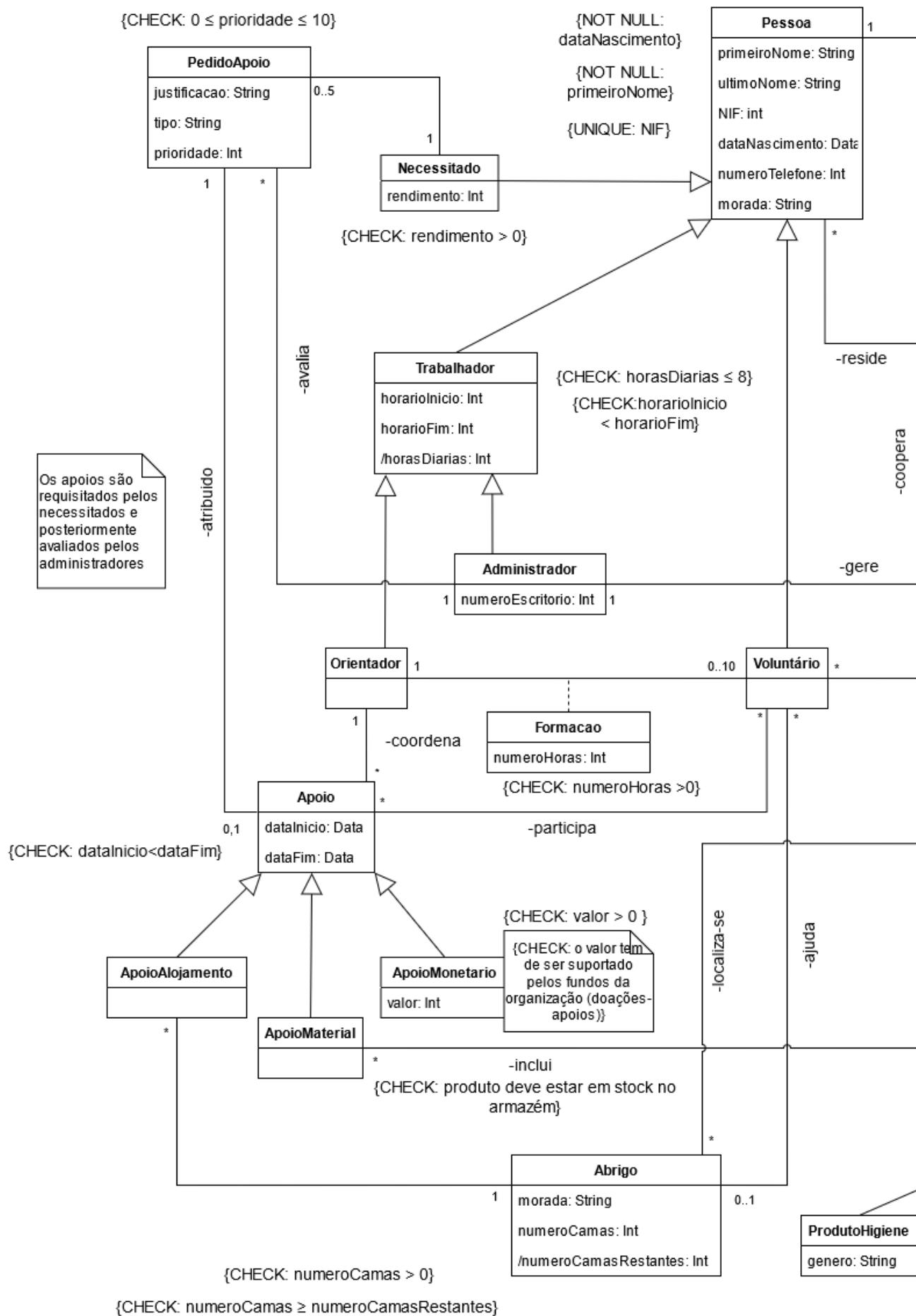
A sustentabilidade da organização é exclusivamente garantida através de apoios por parte de membros solidários da comunidade. São frequentes as doações, quer monetárias quer de bens materiais (alimentares, produtos de higiene e vestuário). De salientar que, no caso de produtos alimentares, a administração exige um prazo mínimo de validade, por forma a garantir a segurança de todos os intervenientes.

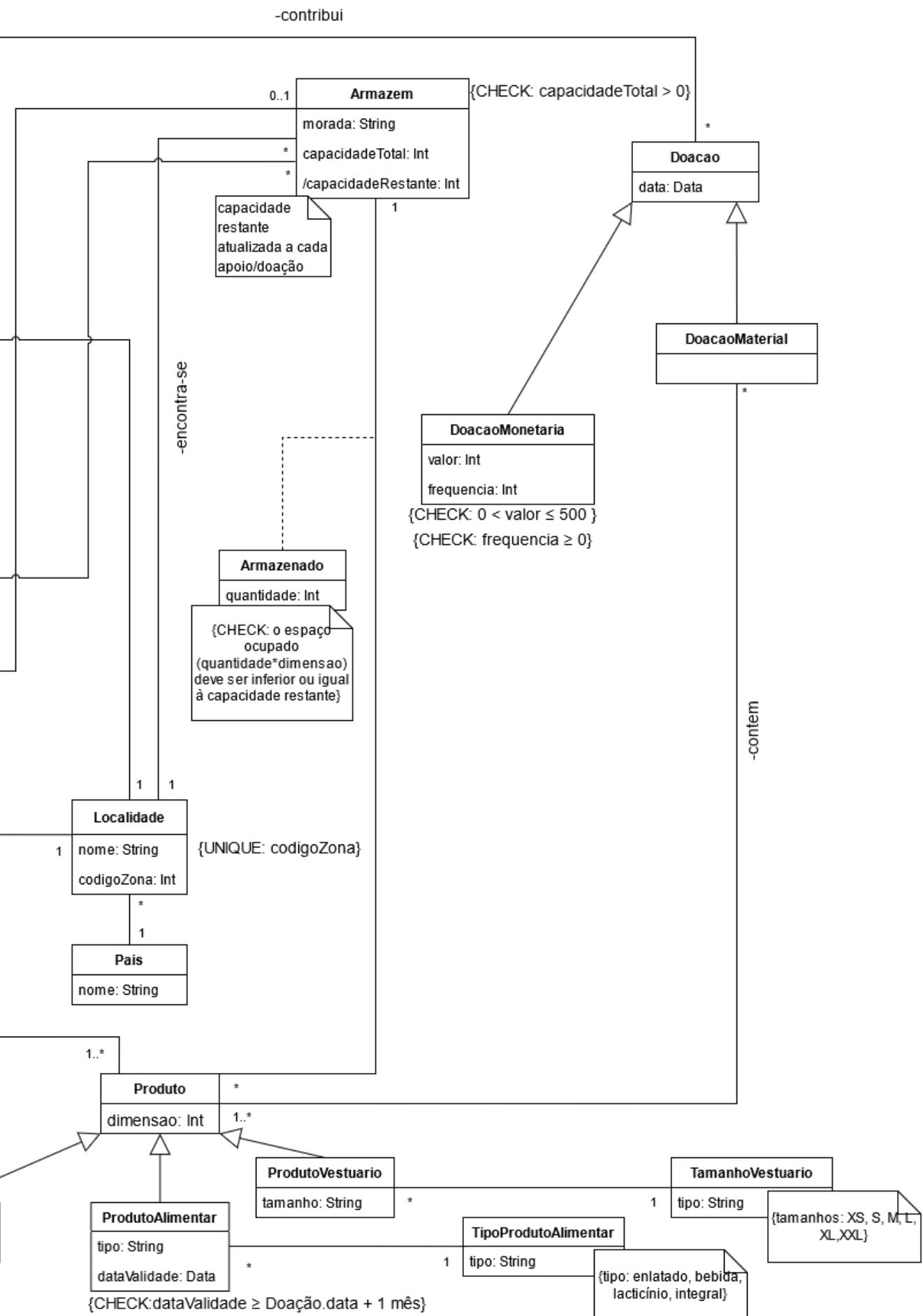
Naturalmente, a possibilidade de doação material pressupõe a possibilidade de armazenamento do produto por parte da organização e as doações monetárias estão legalmente limitadas ao valor de 500€.

São vários os cidadãos altruístas que se dispõem a ajudar regularmente a instituição. Para isso, está prevista a possibilidade de doações monetárias recorrentes com valores e periodicidades flexíveis.

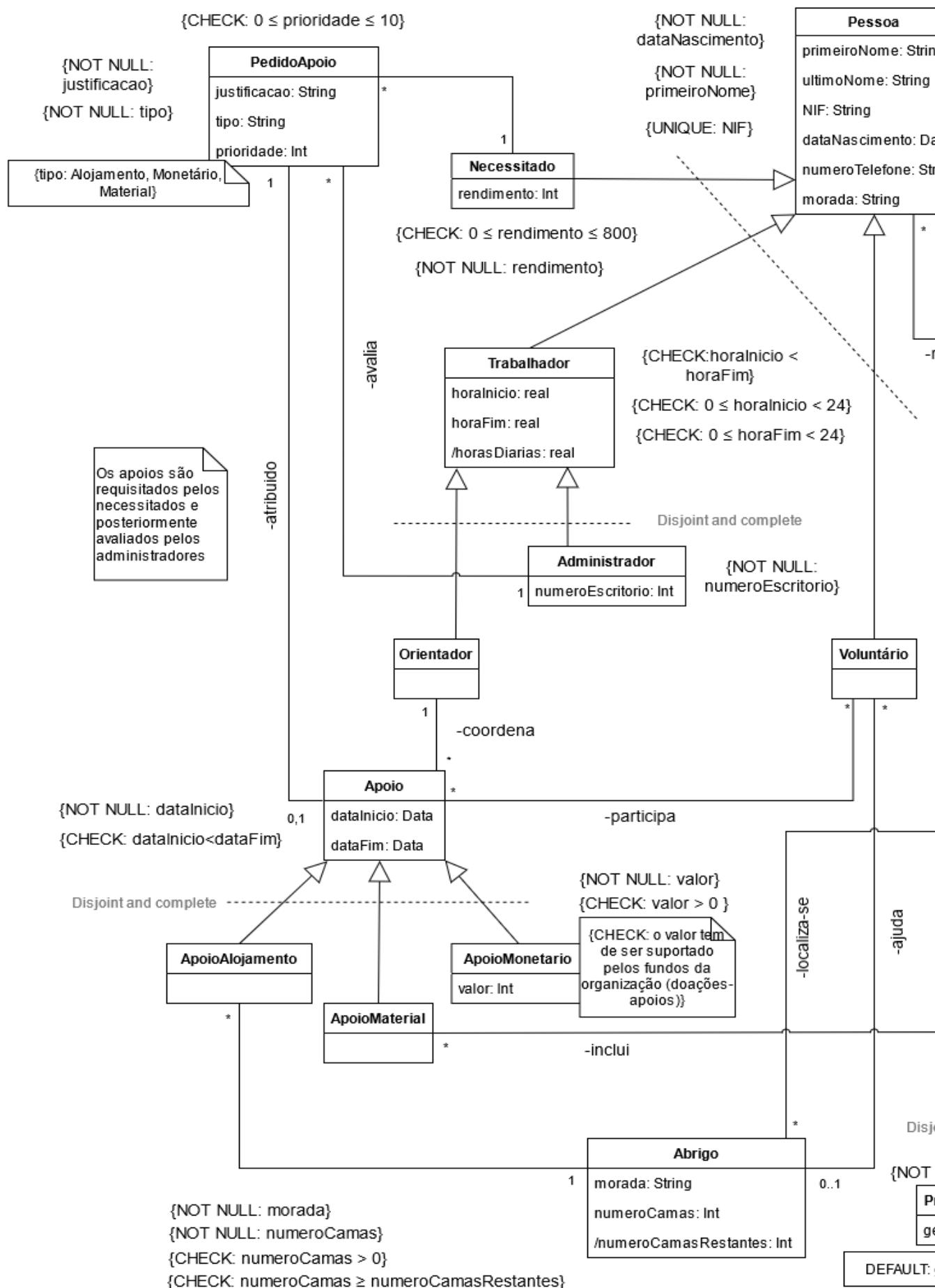
Para efeitos de contabilidade, é necessário armazenar a data de cada doação.

UML

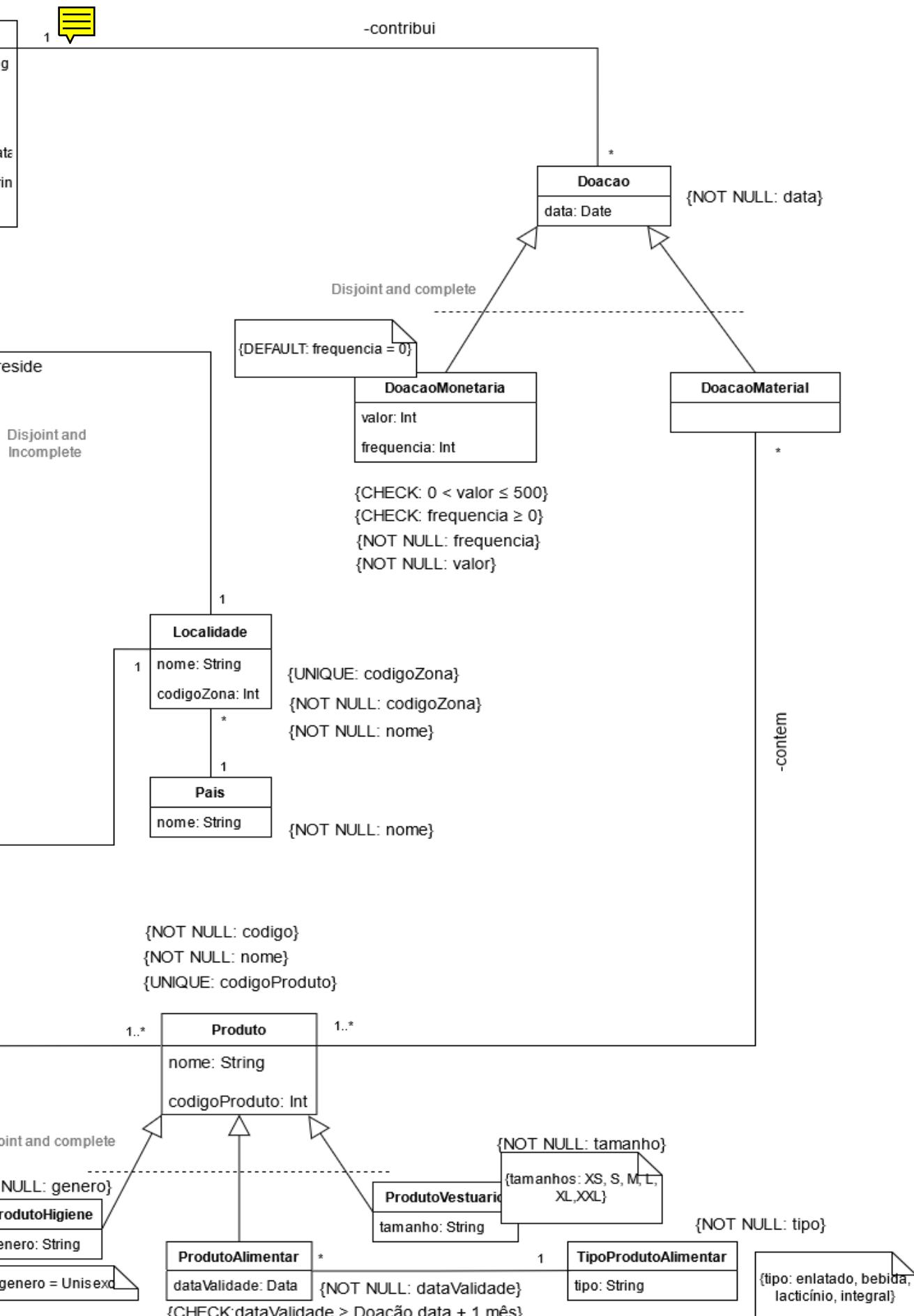




UML Revisto



Por sugestão da professora, foram suprimidas as classes Armazém e Formação, bem como atributo dimensão da classe Produto.



Modelo Relacional

Staff

Na conversão da generalização em Pessoa, foram considerados os três modelos. Por se tratar de uma generalização incompleta, já que não tem em conta os doadores, será necessário manter a superclasse. Dado que esta apresenta ligações e o seu número de atributos é muito superior ao das subclasses, optou-se pela utilização do modelo E/R. Prescindiu-se do modelo OO visto que levaria a relações demasiado repetitivas e obrigaria a replicar todas as relações nas subclasses. A estratégia USE NULLS complicaria as relações dos elementos das subclasses e tornaria as relações esparsas tendo em conta a sua grande diversidade, pelo que se revelou insatisfatória.

- Pessoa(id, primeiroNome, últimoNome, NIF, dataNascimento, númeroTelefone, morada, códigoZona->Localidade)
 - primary key: id;
 - foreign key: códigoZona de Localidade.
 - códigoZona;
- Necessitado(id->Pessoa, rendimento)
 - primary key e foreign key: id,
 - de Pessoa.id;
- Voluntário(id->Pessoa, id->Abrigo)
 - primary key e foreign key: id,
 - de Pessoa.id;

Para converter a generalização de Trabalhador, revelou-se vantajoso o modelo OO tendo em conta que a superclasse não participa em quaisquer relações e o seu número de atributos (2 - horalnício e horaFim) é reduzido. Desconsiderou-se o modelo E/R, tanto por não fazer sentido manter a superclasse como por dificultar futuras implementações obrigando a alterar ou consultar 3 relações. Mesmo com o número reduzido de atributos, o uso de atributos nulos revelou-se inadequado por descontextualizar as relações das subclasses e se prever uma proporção esmagadora de nulos visto que a maior parte dos trabalhadores não pertence à administração.

- Orientador ([id->Pessoa](#), horalnício, horaFim)
primary key e foreign key: id, de Pessoa.id;
- Administrador ([id->Pessoa](#), horalnício, horaFim,
[númeroEscritório](#))
primary key e foreign key: id, de Pessoa.id;



Ações

A existência de poucas relações envolvendo a superclasse e o facto de ser uma generalização completa motivaram a escolha do método OO para representar a generalização em Doação. A utilização de nulos não ofereceria grandes vantagens, quer por retirar contexto às subclasses quer por ser muito danosa na eventualidade de existirem muitas doações materiais. Dado o número reduzido de atributos e os dois tipos de doações serem usados em contextos completamente diferentes, o modelo E/R também não seria conveniente.



- DoaçãoMaterial (id, pessoa->Pessoa, data)
primary key: id;
foreign key: pessoa de Pessoa.id;
- DoaçãoMonetária (id, pessoa->Pessoa, data, valor, frequência)
primary key: id;
foreign key: pessoa de Pessoa.id;



À semelhança de Pessoa, a quantidade reduzida de atributos nas subclasses e a existência de relações envolvendo a superclasse conduziram à utilização da estratégia E/R. Esta abordagem permite-nos manter as relações comuns a todos os tipos de apoio (colaboradores e recursos materiais envolvidos) sem repetições desnecessárias e preservando o contexto próprio de cada apoio sem criar relações esparsas, algo que não seria possível com outros modelos.

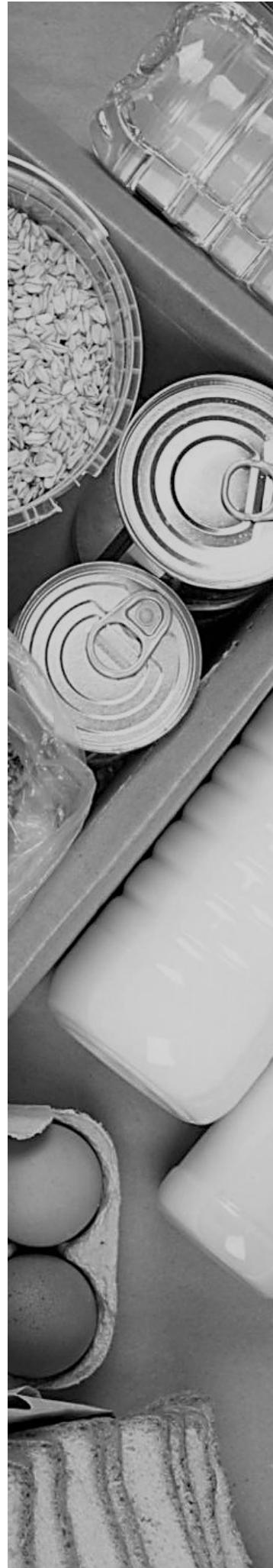
- Apoio(id, dataInicio, dataFim, pedido->PedidoApoio, orientador->Orientador)
 - primary key: id;
 - foreign key: pedido de PedidoApoio.id;
 - foreign key: orientador de Orientador.id;
- ApoioMonetário (id->Apoio, valor)
 - primary key e foreign key: id de Apoio.id;
- ApoioAlojamento (id->Apoio, abrigo->Abrigo)
 - primary key e foreign key: id de Apoio.id;
 - foreign key: abrigo de Abrigo.id;
- ApoioMaterial (id->Apoio)
 - primary key e foreign key: id de Apoio.id;

Produtos

De forma análoga às restantes generalizações, a generalização foi convertida usando E/R.

Tal como anteriormente, a utilização de nulos não teria sido oportuna por retirar elementos do seu contexto e tornar a relação exclusiva entre ProdutoAlimentar e TipoAlimentar difícil de assegurar. O método OO obrigaria à criação de relações individuais para as subclasses, o que não oferece vantagens aparentes.

- Produto (código, nome)
primary key: código;
- ProdutoHigiene (id->Produto, género)
primary key e foreign key: código de Produto.código;
- ProdutoVestuário (id->Produto, tamanho)
primary key e foreign key: código de Produto.código;
- ProdutoAlimentar (id->Produto, dataValidade, tipo->TipoAlimentar)
primary key e foreign key: código de Produto.código;
foreign key: tipo de TipoAlimentar.id
- **TipoAlimentar (id, tipo)**
primary key: id;





Classes individuais

- Localidade (código, códigoPais-> País, nome)
 - primary key: código;
 - foreign key: códigoPais de País.código;
- País (código, nome)
 - primary key: código;
- PedidoApoio (id, justificação, tipo, prioridade, avaliador->Administrador, pedinte->Necessitado)
 - primary key: id;
 - foreign key: avaliador de Administrador.id;
 - foreign key: pedinte de Necessitado.id;
- Abrigo(id, morada, númeroCamas, códigoZona->Localidade)
 - primary key: id;
 - foreign key: códigoZona de Localidade.códigoZona;

Outras Relações

- DoaçãoMaterialContemProduto (doação->Doação-Material, produto->Produto)
foreign key: doação de DoaçãoMaterial.id;
foreign key: produto de Produto.código;
composite primary key: doação e produto;
- ProdutoIncluiApoyoMaterial (apoio->Apoyo,
produto->Produto) 
foreign key: apoio de Apoyo.id;
foreign key: produto de Produto.código;
composite primary key: produto e apoio;
- VoluntárioParticipaApoyo (voluntário->Voluntário,
apoio->Apoyo)
foreign key: voluntário de Voluntário.id;
foreign key: apoio de Apoyo.id;
composite primary key: voluntário e apoio;



Restrições

Pessoa

- primeiroNome: para efetuar uma identificação básica, é necessário conhecer o primeiro nome de uma pessoa (NOT NULL);
- dataNascimento: de forma a personalizar o atendimento é necessário conhecer a data de nascimento de uma pessoa (NOT NULL);
- **codigoZona**: para alocar recursos de forma eficiente, é necessário ter em conta a zona aproximada de residência da pessoa (NOT NULL); 
- como legalmente estipulado, o NIF de uma pessoa é único (UNIQUE);
- id é chave primária (key constraint, PRIMARY KEY);

Necessitado

- rendimento: de forma a garantir o apoio correto e otimizar a alocação de recursos, é essencial conhecer a situação financeira do necessitado (NOT NULL);
- O rendimento não poderá ser negativo, podendo ser zero nos mais desfavorecidos (CHECK: rendimento ≥ 0);
- Para garantir que só as pessoas em extrema necessidade são auxiliadas, não serão admitidas pessoas com rendimentos superiores a 800€ (CHECK: rendimento ≤ 800);
- id é chave primária (key constraint, PRIMARY KEY); 

Voluntário

- id é chave primária (key constraint, PRIMARY KEY);



Orientador

- Para gerir turnos e responsabilidades, é necessário conhecer os horários de trabalho dos orientadores (horalnico NOT NULL e horaFim NOT NULL);
- Evidentemente, a horaFim tem de ser superior à horalnicio (CHECK (horaFim > horalnicio));
- Naturalmente, as horas terão de pertencer ao intervalo válido [0, 24[. (CHECK (0 <= horalnicio and horalnicio < 24) e CHECK (0 <= horaFim and horaFim < 24));
- id é chave primária (key constraint, PRIMARY KEY);



Administrador

- Para gerir turnos e responsabilidades, é necessário conhecer os horários de trabalho dos administradores (horalnico NOT NULL e horaFim NOT NULL);
- Evidentemente, a horaFim tem de ser superior à horalnicio (CHECK (horaFim > horalnicio));
- Naturalmente, as horas terão de pertencer ao intervalo válido [0, 24[. (CHECK (0 <= horalnicio and horalnicio < 24) e CHECK (0 <= horaFim and horaFim < 24));
- númeroEscritório: cada Administrador necessita de um escritório para trabalhar (NOT NULL);
- id é chave primária (key constraint, PRIMARY KEY);



DoacaoMaterial

- data: a data de entrega da doação deve ser mantida em registo, ajudando a estabelecer limites de prazo de validade para produtos alimentares (NOT NULL);
- id é chave primária (key constraint, PRIMARY KEY);



DoacaoMonetaria

- data: para efeitos de contabilidade, a data de doação tem que ser registada. No caso de doações recorrentes, é útil para notificações. (NOT NULL);
- valor: uma doação monetária é obrigatoriamente caracterizada pelo montante doado (NOT NULL);
- frequência: para planeamento de atividades, importa ter em conta a quantidade de doações regularmente esperadas. (NOT NULL);
- limiteMonetário: os montantes serão naturalmente positivos e não poderão exceder os 500€ por restrições legais (CHECK ($0 < \text{valor}$ and $\text{valor} \leq 500$));
- frequenciaVálida: a frequência poderá ser positiva ou nula no caso de doações únicas (CHECK ($\text{frequência} \geq 0$));
- id é chave primária (key constraint, PRIMARY KEY);



Apoio

- dataInício: ao contrário da dataFim, a data de início de apoio é necessária para o cálculo de futuros apoios ou alocação de recursos a novos pedidos de ajuda (NOT NULL);
- pedido: todos os apoios são atribuídos após um pedido. Para poder consultar informação acerca do necessitado, este atributo terá sempre que corresponder a um elemento da tabela de necessitados, não podendo ser nulo (NOT NULL);
- orientador: todos os apoios são obrigatoriamente coordenados por um orientador (NOT NULL);
- Para corresponder a uma situação coerente a dataFim tem de ser superior à dataInício (CHECK ($\text{dataFim} > \text{dataInício}$));
- id é chave primária (key constraint, PRIMARY KEY);



ApoioMonetario

- valor: será sempre necessário conhecer o montante atribuído em cada apoio (NOT NULL);
- valorPositivo: embora seja diferente consoante a necessidade, o apoio atribuído será sempre positivo (CHECK (valor > 0));
- id é chave primária (key constraint, PRIMARY KEY e FOREIGN KEY);

ApoioAlojamento

- abrigo: será necessário conhecer em que abrigo o necessitado ficou alojado (NOT NULL);
- id é chave primária (key constraint, PRIMARY KEY e FOREIGN KEY);

ApoioMaterial

- id é chave primária (key constraint, PRIMARY KEY e FOREIGN KEY);

Produto

- nome: um produto, por questões armazenamento necessita de estar etiquetado com o seu nome (NOT NULL);
- código é chave primária (key constraint, PRIMARY KEY);

ProdutoHigiene

- género: para que seja entregue às pessoas adequadas, este atributo tem de ser preenchido (NOT NULL). Na eventualidade não ter género, é etiquetado como ‘Unisexo’;
- id é chave primária (key constraint, PRIMARY KEY e FOREIGN KEY);

ProdutoVestuario

- tamanho: o tamanho do vestuário é importante para ser corretamente armazenado e posteriormente oferecido (NOT NULL).
- tamanhoExistente: o tamanho das peças terá de corresponder a um dos tamanhos padronizados (CHECK (tamanho LIKE ‘XS’ or tamanho LIKE ‘S’ or tamanho LIKE ‘M’ or tamanho LIKE ‘L’ or tamanho LIKE ‘XL’ or tamanho LIKE ‘XXL’));
- id é chave primária (key constraint, PRIMARY KEY e FOREIGN KEY);

ProdutoAlimentar

- dataValidade: para assegurar a saúde de todos e otimizar a gestão dos recursos, a data de validade do produto tem de ser armazenada (NOT NULL);
- tipo: para acondicionar corretamente os produtos, será necessário conhecer o seu tipo (NOT NULL);
- id é chave primária (key constraint, PRIMARY KEY e FOREIGN KEY);

TipoAlimentar

- tipo: descreve o tipo de alimento, tem de existir de forma a ter poder ser interpretado facilmente pelas pessoas correspondentes (NOT NULL);
- id é chave primária (key constraint, PRIMARY KEY);

Localidade

- codigoPais: todas as localidades estão, por definição, inseridas num país (NOT NULL);
- nome: como é a forma habitual de identificar uma localidade, a sua designação é obrigatória (NOT NULL);
- codigoZona é chave primária (key constraint, PRIMARY KEY);

País

- nome: por ser a forma habitual de identificar um país, a sua designação é obrigatória (NOT NULL);
- código é chave primária (key constraint, PRIMARY KEY);



PedidoApoio

- justificação: a justificação é o principal critério de seriação dos pedidos, sendo imprescindível para uma avaliação eficaz (NOT NULL);
- tipo: é necessário que o pedido de apoio seja concreto relativamente ao tipo de ajuda solicitada para que o seu tratamento seja agilizado (NOT NULL);
- tipoValido: o tipo de apoio requisitado deve corresponder a um dos fornecidos pela organização (CHECK (tipo LIKE 'Alojamento' or tipo LIKE 'Material' or tipo LIKE 'Monetário'));
- prioridade: o avaliador necessita de classificar o pedido de forma a ser inserido na fila de prioridade já estipulada (NOT NULL);
- avaliador: todos os pedidos são avaliados por um administrador (NOT NULL);
- pedinte: todos os pedidos são efetuados por um necessitado, acerca de quem poderá ser necessário consultar informações (NOT NULL);
- limitesPrioridade: a prioridade é estabelecida seguindo um padrão da própria instituição, que a restringe ao intervalo [0,10], 0 se o pedido for liminarmente rejeitado (CHECK (prioridade >= 0 and prioridade <= 10));
- id é chave primária (key constraint, PRIMARY KEY);

Abrigo

- morada: informação vital para identificar o abrigo em questão e adequar a atribuição de pessoas em função da proximidade ao mesmo (NOT NULL);
- númeroCamas: a capacidade máxima do abrigo permite realizar cálculos apropriados, garantindo desta forma que não são distribuídos grupos numerosos de necessitados para abrigos com instalações incapazes de os acolher (NOT NULL);
- códigoZona: para facilitar a ordenação de abrigos, de acordo com a sua distância a um dado local (em plataformas destinadas a tal), o código da localidade faz parte da informação obrigatória (NOT NULL);
- númeroCamasPositivo: as instalações do abrigo precisam de poder acolher pelo menos um necessitado (CHECK (númeroCamas > 0));
- id é chave primária (key constraint, PRIMARY KEY);



Dependências Funcionais

Pessoa

{id} -> {primeiroNome, últimoNome, NIF, dataNascimento, númeroTelefone, morada, códigoZona}
{NIF} -> {id, primeiroNome, últimoNome, dataNascimento, númeroTelefone, morada, códigoZona}

Administrador

{id} -> {horarioInicio, horaFim, numeroEscritorio}
{horarioInicio, horaFim} -> {tempoDeTrabalho}

Orientador

{id} -> {horarioInicio, horaFim}
{horarioInicio, horaFim} -> {tempoDeTrabalho}

Necessitado

{id} -> {rendimento}

Apoio

{id} -> {dataInicio, dataFim, pedido, orientador}
{pedido} -> {dataInicio, dataFim, orientador, id}

ApoioMonetario

{id} -> {valor}

ApoioAlojamento

{id} -> {abrigos}

PedidoApoio

{id} -> {justificação, tipo, prioridade, avaliador}

DoaçãoMonetária

{id} -> {pessoa, data, valor, frequência}

DoaçãoMaterial

{id} -> {pessoa, data}

Produto

{código} -> {nome}

ProdutoHigiene

{id} -> {género}

ProdutoAlimentar

{id} -> {dataValidade, tipo}

ProdutoVestuário

{id} -> {tamanho}

TipoAlimentar

{id} -> {tipo}

{tipo} -> {id}

Localidade

{códigoZona} -> {nome, códigoPais}

País

{código} -> {nome}

Abrigo

{id} -> {morada, númeroCamas, códigoZona}

{morada, códigoZona} -> {id, númeroCamas}



Formas Normais

Forma Normal de Boyce-Codd (BCNF)

Diz-se que uma relação se encontra na BCNF se e só: se para cada dependência funcional não trivial $\bar{A} \rightarrow \bar{B}$, \bar{A} é uma (super)chave.

Terceira Forma Normal (3NF)

Diz-se que uma relação se encontra na 3NF se e só:

1. se encontra na 2NF e todos os atributos não primos são funcionalmente dependentes da chave de forma não transitiva

OU

2. para cada dependência funcional não trivial $\bar{A} \rightarrow \bar{B}$:

a. \bar{A} é uma (super)chave;

OU

b. \bar{B} consiste exclusivamente de atributos primos

Violações à BCNF ou 3NF

Tendo em conta que todas as dependências funcionais na BCNF também se encontram na 3NF (ver ponto 2a), justificar-se-á apenas o primeiro caso.

Nas duas dependências funcionais $\{horalnicio, horaFim\} \rightarrow \{tempoDeTrabalho\}$ encontradas em Orientador e Administrador, apesar de existir uma violação da BCNF/3NF (o lado esquerdo não é uma chave), o atributo dependente é derivado e portanto considerou-se que as relações não são suscetíveis de decomposição.

As restantes dependências têm do seu lado esquerdo uma chave (candidata ou superchave), o que nos leva a concluir que se encontram todas na BCNF e consequentemente na 3NF.

Para as seguintes dependências funcionais, julgou-se necessário evidenciar o contexto em que se enquadram de modo a justificar a razão pela qual se consideram chaves os conjuntos do lado esquerdo:

Abrigo: $\{morada, codigoZona\} \rightarrow \{id, numeroCamas\}$. Considera-se impossível a hipótese de existirem dois ou mais abrigos com a mesma morada e código de zona.

TipoAlimentar: $\{tipo\} \rightarrow \{id\}$. Esta classe equipara-se a uma enumeração, no sentido em que a cada id corresponde apenas um tipo de produto alimentar e vice-versa.

Apoio: $\{pedido\} \rightarrow \{dataInício, dataFim, orientador, id\}$. Um apoio requer a aprovação de um pedido por um determinado administrador. Como a cada apoio só corresponde um pedido, este pode ser utilizado como chave da relação.

