Universidade do Minho
Escola de Engenharia
Licenciatura em <licenciatura>

Unidade Curricular de
Bases de Dados

Ano Lectivo de 2014/2015

Associação Humanitária Habitat
Grupo 24

R. Oliveira (67661), R. Camposinhos (72625),
J. Pereira (67680), T. Ferreira (67701)

Janeiro, 2015

Data de Recepção	
Responsável	
Avaliação	
Observações	

Associação Humanitária Habitat

Grupo 24

R. Oliveira (67661), R. Camposinhos (72625),

J. Pereira (67680), T. Ferreira (67701)

Janeiro, 2015

Resumo

Neste relatório apresenta-se o projeto académico da unidade curricular de base de dados. O projeto consiste na criação de uma base de dados relacional, seguindo a metodologia proposta por Connolly & Begg (2002). A base de dados foi desenvolvida para um contexto de aplicação real, nomeadamente para uma instituição de solidariedade social, a Habitat, que tem por objetivo combater a pobreza habitacional, construindo ou recuperando habitações para famílias carenciadas. Todas as fases de desenvolvimento da base de dados encontram-se devidamente documentadas, em particular o levantamento e análise de requisitos, a elaboração, descrição e validação do modelo conceptual e a conversão do mesmo para o modelo lógico, a validação do modelo lógico e por fim a conversão do modelo lógico para o modelo físico, com a consequente implementação. Em anexo encontram-se importantes elementos como os diagramas dos modelos conceptual e lógico, o respetivo dicionário de dados, bem como a script de criação do modelo físico.

Área de Aplicação: Desenho e arquitetura de Sistemas de Bases de Dados.
Palavras-Chave: Habitat, Bases de Dados Relacionais, Modelo Conceptual, Modelo Lógico, Modelo Físico, Entidades, Relacionamentos, Atributos, Chaves, Tabelas, Transações.

Índice

Anexos

Índice de Figuras

Índice de Tabelas

1. Introdução

1.1. Contextualização

O presente caso de estudo foi desenvolvido para a Associação Humanitária Habitat – *Habitat for Humanity Portugal*[1], que é uma organização cristã sem fins lucrativos dedicada à eliminação da pobreza habitacional.

O trabalho foi elaborado em contexto académico e teve como principal objetivo criar uma base de dados, que serviu de base a uma aplicação informática desenvolvida no âmbito da unidade curricular de Desenvolvimento de Sistemas de Software.

A BD desenvolvida segue a metodologia relacional proposta por Connolly & Begg (2002).

1.2. Apresentação do Caso de Estudo

A Habitat tem como principal objetivo ajudar famílias carenciadas a construir, ou reabilitar, habitações. A instituição divide-se em quatro órgãos fundamentais:

- Direção – órgão máximo, responsável pela aprovação dos processos;
- Comissão de fundos – responsável pela angariação de donativos e voluntários;
- Comissão de famílias – responsável pelo acompanhamento e apreciação das candidaturas das famílias;
- Comissão de construção – responsável pela gestão e realização das empreitadas.

Todos os dados relativos às famílias, voluntários, donativos e empreitadas necessitam de ser preservados pela instituição de forma estruturada, de forma a manter o seu bom funcionamento e uma capacidade de resposta eficaz.

[1] http://www.habitat.pt/

1.3. Motivação e Objetivos

Com base no anteriormente exposto, o principal objetivo do presente trabalho é desenvolver uma base de dados com toda a informação necessária para a Habitat.

Para além do objetivo primário enunciado, a base de dados a desenvolver deverá possuir algumas características fundamentais, tais como:

- Consistência;
- Persistência;
- Reduzida redundância;
- Escalabilidade;
- Facilidade de manipulação e eventual adaptação futura.

Seguidamente apresenta-se, de uma forma sintética, a fundamentação e principais objetivos da base de dados a desenvolver:

- Armazenar toda a informação das famílias necessária para as candidaturas, assim como sobre o seu acompanhamento após a entrega da nova habitação;
- Armazenar toda a informação relevante dos voluntários, nomeadamente os dados gerais de identificação, contactos e registo de tempo de voluntariado despendido para a Habitat;
- Armazenar toda a informação relevante dos doadores, nomeadamente os seus dados de identificação e histórico de donativos realizado;
- Armazenar dados relacionados com as empreitadas, designadamente o seu custo e duração, as tarefas realizadas e planeadas, entre outros dados genéricos.
- Identificar que donativos contribuíram para determinada empreitada.
- Identificar que voluntários contribuíram para determinada empreitada.

Para além da motivação óbvia e inerente ao currículo académico, a realização do presente trabalho teve como fonte extra de motivação a possibilidade de participar num caso com um contexto real e a possibilidade de apoiar uma instituição com uma missão muito louvável. A aplicação da metodologia relacional lecionada nas aulas da unidade curricular de BD, com implementação do ciclo completo de uma BD, constitui também um fator extra de motivação.

1.4. Estrutura do Relatório

Como já referido, após o presente capítulo introdutório, no capítulo 2 é descrita a metodologia utilizada para a identificação dos requisitos, bem como a sua descrição e numeração. No capítulo 3 é feita uma breve descrição e caracterização dos perfis de utilização. Os três capítulos sucessivos são os capítulos principais do relatório, que detalham a metodologia desenvolvida para a conceção dos modelos conceptual, lógico e físico. Por fim, apresentam-se

as conclusões e desenvolvimentos futuros. Em anexo apresentam-se, entre outros elementos, os diagramas do modelo conceptual e lógico, assim como o respetivo dicionário de dados.

2. Levantamento e Análise de Requisitos

O levantamento e análise de requisitos foram realizados através de conferência com o representante da Habitat. Para tal foram organizadas duas sessões de pergunta e resposta, que decorreram nos dias 22/10/2014 e 13/11/2014.

Ao longo das várias aulas foram também expostas dúvidas e questões ao Prof. Orlando Belo, que serviu de interlocutor com a Habitat.

Foram também disponibilizados os formulários relativos às candidaturas das famílias e registo de voluntários, que serviram de base à definição da BD. Excertos dos referidos formulários encontram-se expostos na Figura 1 e Figura 2.

Durante o período de levantamento não foi possível endereçar, por escrito, à Habitat questões específicas que teriam sido úteis para completar o processo, designadamente:

Questões a responder por cada perfil de utilizador

- Qual é o seu cargo na Habitat? E com quem contacta usualmente?
- Que tipo de tarefas realiza habitualmente num dia de trabalho?
- Com que tipo de dados costuma trabalhar?
- Que tipo de relatórios/listagens costuma utilizar?
- Que tipo de dados necessita acompanhar?

Associação Humanitária Habitat
Habitat for Humanity®
Braga | Portugal

COORDENAÇÃO DE VOLUNTARIADO

"Construindo casas, construindo esperança"

FICHA DE INSCRIÇÃO

NOME			
DATA NASCIMENTO		**PROFISSÃO**	
MORADA			
CÓDIGO POSTAL		**LOCALIDADE**	
TELEFONE		**TELEMÓVEL**	
E-MAIL	@		
HABILITAÇÕES ACADÉMICAS			
CONHECIMENTOS LINGUÍSTICOS			
FORMAÇÃO COMPLEMENTAR			
EXPERIÊNCIA VOLUNTARIADO			
GOSTARIA DE PARTICIPAR NAS ACTIVIDADES DE ALGUMA DAS NOSSAS COMISSÕES OU ACTIVIDADES? POR FAVOR CITAR QUAIS	☐ Direcção & Gestão ☐ Conselho Fiscal ☐ Comissão de Angariação de Fundos & Relações Públicas ☐ Comissão de Apoio e Selecção Famílias ☐ Comissão de Construção e Gestão de Projectos ☐ Comissão de Desenvolvimento de Voluntariado ☐ Comissão de Finanças ☐ Actividades de Angariação de Fundos ☐ Actividades de Apoio Administrativo ☐ Actividades de Apoio Voluntários Estrangeiros & Global Village ☐ Actividades com Entidades Religiosas ☐ Actividades com Informática & Media | Publicidade & Eventos ☐ Voluntariado Internacional – Global Village		
CONHECIMENTOS CONSTRUÇÃO		**GOSTARIA DE TRABALHAR NA OBRA JUNTO A OUTROS VOLUNTÁRIOS?**	
DISPONIBILIDADE DE TEMPO			
COMO CONHECEU A HABITAT?		**DESEJARIA RECEBER INFORMAÇÕES SOBRE A HABITAT?**	

Obrigado pelo seu interesse no trabalho da *Habitat For Humanity*. Agradecemos o vosso interesse em inscrever-se e tornar-se parceiro na missão da Habitat, na luta por uma habitação digna para famílias em necessidade no mundo. Seu esforço significa um grande instrumento para manter a sustentabilidade do programa de construção desta filial da Habitat em Portugal. Muito obrigado!

Associação Humanitária Habitat
Praça Conde do Agrolongo, 35, 2º - Sala 8
4704-524 Braga

Contacto: **João Cruz**
habitat@hfhportugal.org
Telefone: 253 204 280
Fax: 253 204 287

Braga, __,_____,20__

Figura 1 – Ficha de inscrição dos voluntários

Habitat for Humanity Portugal

Ficha de Inscrição

Candidato/a:

Nome _____
Estado civil _____
Data de nascimento _____
Escolaridade _____
Profissão _____
Naturalidade _____
Nacionalidade _____
Morada _____
Telefone _____

Questionário

1. A sua casa é:

casa própria □ Qual o valor da prestação? €
casa arrendada □ Qual o montante da renda? €
casa cedida □
outro tipo □ Qual?

2. Quais são os principais problemas da casa onde vive actualmente?

_____
_____

3. O que espera da Associação Humanitária Habitat?

- Ajuda para a construção de uma Habitação em terreno próprio ou doado?..... □
- Obras de reparação/beneficiação da casa actual..... □

4. Durante a realização da obra, terá onde ficar? □ Sim □ Não

5. Qualquer obra de construção de raiz ou reparação implica custos e portanto terá de ser paga à Associação Humanitária Habitat. Está disposto a contribuir com uma mensalidade?

_____

Habitat for Humanity® Portugal
Praça Conde de Agrolongo, nº35 – 2º, 4700-312 Braga, Portugal
Tel: 253 204 280
E-mail: habitat@hfhportugal.org | www.habitat.pt | www.facebook.com/hfhportugal

Figura 2 – Ficha de inscrição de famílias (excerto).

2.1. Requisitos

Nos pontos seguintes enumeram-se os requisitos identificados para a presente BD:

1. A BD deverá guardar informação sobre os voluntários, famílias, doadores e projetos.
2. Os funcionários de cada comissão registam respetivamente os referidos dados, podendo consultar a totalidade dos dados existentes.

Sobre voluntários:

3. Existem dois tipos de voluntários: individuais e equipas organizadas.
4. Os dados necessários para o registo dos voluntários deverão obedecer à ficha de inscrição disponibilizada pela Habitat (Figura 1).
5. Os voluntários têm um ramo de atividade.

6. Os voluntários podem realizar várias tarefas e participar em mais do que uma equipa.
7. As tarefas realizadas pelos voluntários deverão ser registadas: tempo despendido, data, descrição da tarefa e projeto.
8. Os registos das tarefas realizadas em equipa deverão ser separados dos registos individuais.

Sobre donativos:

9. Os dados necessários para os doadores são semelhantes aos dos voluntários, com acréscimo do NIF (número de contribuinte) para a emissão dos recibos.
10. Os doadores podem ser pessoas coletivas ou individuais. Existem também doadores que são parceiros da Habitat.
11. Tal como os voluntários, também os doadores têm um ramo de atividade.
12. Os doadores podem realizar vários donativos.
13. Os donativos poderão ser em dinheiro ou em espécie (por exemplo: materiais, equipamentos, serviços).
14. Os donativos deverão ser associados, na sua totalidade ou em parte, aos projetos realizados pela Habitat.
15. Deverão ser registadas as transferências dos donativos para os projetos (data e a quantidade de um donativo utilizada).
16. Existem eventos de angariação de donativos. Deverá ser possível identificar que donativos são provenientes de um dado evento.
17. Um doador pode ser simultaneamente um voluntário.

Sobre os projetos de construção:

18. Um projeto deverá possuir uma designação, um orçamento base, uma estimativa de prazo, uma data de início e um conjunto de tarefas planeadas. A data de fim da obra também deve ser registada, para efeitos contabilização do período de garantia.
19. As tarefas realizadas pelos voluntários deverão corresponder, sempre que possível, às tarefas planeadas, para que seja possível comparar e acompanhar o progresso da obra. No entanto, poderão ser realizadas tarefas não planeadas.
20. Os projetos resultam de candidaturas aprovadas, devendo ser possível identificar qual a candidatura e família para o qual é realizado o projeto.

Sobre as famílias:

21. Os dados necessários para o registo das famílias deverão obedecer à ficha de inscrição disponibilizada pela Habitat (Figura 2).
22. Uma família tem um representante e pode ter vários elementos do seu agregado familiar.
23. Uma família pode realizar mais do que uma candidatura, mas só pode ter um projeto aprovado.
24. Uma família tem um rendimento anual bruto, sendo este um dos dados essenciais para a apreciação das candidaturas.

25. Uma candidatura pode ser classificada como aprovada, não aprovada, não aceite (pela família), ou pendente.
26. A aprovação, ou não, da candidatura é realizada pela direção da Habitat.
27. Após a aprovação de uma candidatura, e sua aceitação por parte da família, é acordada uma prestação, a ser paga pela família à Habitat após a conclusão da obra. Essa prestação pode ser posteriormente alterada, sendo necessário guardar o seu histórico.
28. Após a entrega das habitações, a Habitat realiza um acompanhamento das famílias, que é materializado em relatórios.

2.2. Queries

Nos pontos seguintes enumeram-se alguns exemplos das consultas (*queries)* que se esperam que sejam mais frequentes. Pese embora não seja um levantamento exaustivo, estas consultas foram utilizadas para validação das várias etapas de desenvolvimento da BD (ver capítulos "Validação com Transações do Utilizador").

1. Listar os projetos para os quais um doador contribui.
2. Listar os projetos para os quais um voluntário trabalhou individualmente.
3. Listar os donativos alocados a um determinado projeto.
4. Identificar a família a que corresponde um determinado projeto.
5. Listar os relatórios realizados para uma dada família.
6. Listar o número de doadores por atividade.

3. Caracterização dos Perfis de Utilização

No presente capítulo apresenta-se uma breve descrição da utilização do SGBD, pelo menos numa fase inicial.

A utilização terá quatro perfis distintos, como descrito seguidamente:

- Um perfil de administração, responsável por realizar todas as operações de administração que são vedadas aos utilizadores comuns.
- Um perfil relativo à Comissão de Fundos, que será responsável por introduzir todos os dados relativos aos voluntários, doadores e angariação de donativos. Este perfil poderá consultar a informação dos restantes perfis, no entanto não poderá introduzir nem alterar dados fora do âmbito da Comissão de Fundos.
- Um perfil relativo à Comissão de Famílias, que será responsável pela introdução de todos os dados relativos às famílias, candidaturas e relatórios. Este perfil poderá consultar a informação dos restantes perfis, no entanto não poderá introduzir nem alterar dados fora do âmbito da Comissão de Famílias.
- Um perfil relativo à Comissão de Construção, que será responsável pela introdução de todos os dados relativos aos projetos e tarefas realizadas pelos voluntários. Será também responsável por determinar a alocação dos donativos aos projetos. Este perfil poderá consultar a informação dos restantes perfis, no entanto não poderá introduzir nem alterar dados fora do âmbito da Comissão de Construção.

O SGBD operará numa rede local de reduzida dimensão que suportará os seguintes equipamentos:

- 1 Servidor;
- 2 Desktops;
- 2 Laptops;
- 1 Router.

Espera-se que o volume de dados seja de reduzida dimensão. De acordo com o site oficial da organização, “desde a sua fundação em 1996, já apoiou mais de 60 famílias a terem uma casa digna, ajudando cerca de 250 pessoas”. Pelo exposto, espera-se que sejam criados cerca de 3 a 4 novos projetos por ano. No capítulo relativo ao Modelo Físico será desenvolvido este tema com maior detalhe.

4. Modelo Conceptual

No presente capítulo descrevem-se todas as etapas de base para a definição do modelo conceptual.

As etapas abaixo descritas culminaram na elaboração de um diagrama ER, que se apresenta no Anexo – Diagrama .

O diagrama foi desenvolvido com recurso à ferramenta TerraER[2], recorrendo a uma notação próxima da definida por Chen (1976) e com as seguintes particularidades:

- Relacionamento com traço duplo => relacionamento obrigatório;
- Relacionamento com traço simples => relacionamento opcional;
- Atributo com linha tracejada => atributo derivado;
- Atributo com traço duplo => atributo multi-valor;
- Atributo com texto sublinhado => chave primária.

De forma a facilitar a leitura do diagrama foram somente considerados os atributos principais e suficientes para uma caracterização geral do problema.

4.1. Identificação de Entidades

No modelo relacional desenvolvido foram identificadas 13 entidades. A identificação das entidades foi realizada de forma iterativa. Numa primeira fase foram identificados as entidades mais óbvias, associadas por exemplo a substantivos ou sintagmas nominais. Algumas foram identificadas de forma quase trivial na própria fundamentação. Posteriormente procedeu-se a um refinamento contínuo, procurando cumprir todos os requisitos definidos.

Pelo exposto, as entidades principais e que foram imediatamente identificadas são:

- Família – necessária para armazenar dados sobre os elementos da família (req. 21);
- Projeto – necessária para os registos dos projetos (req. 18);
- Funcionário – através do req. 2;
- Indivíduo (Doador e/ou Voluntário) – numa fase inicial foram definidas duas entidades separadas, no entanto, tendo em conta que os dados de identificação são semelhantes e que um voluntário pode ser simultaneamente doador, optou-se por

[2] http://www.terraer.com.br/

juntar as duas entidades em uma só. A definição desta entidade responde aos requisitos req. 4, req. 9 e req. 17.

Numa segunda linha de importância surgem as seguintes entidades:

- Candidatura – uma família pode realizar várias candidaturas (req. 23);
- Tarefa – definida devido aos requisitos req. 18 e req. 19;
- Donativo – resposta ao conjunto de requisitos req. 12 a req. 15;
- Evento – necessária para o req. 16.
- Equipa – de acordo com req. 3 e req. 8.
- Relatório – resulta da necessidade definida no req. 28.

Por fim, foram identificados outras duas entidades, que resultaram de opções de implementação:

- Questão – os formulários de candidatura das famílias possuem várias questões; a definição da entidade Questão permitirá uma pesquisa por tipo de questão bem como adicionar eventuais futuras questões. A alternativa a esta opção seria considerar todas as questões como atributos da entidade Candidatura. Entende-se que esta opção conduz a uma melhor estruturação da BD e a uma maior sustentabilidade da mesma.
- Tipo de Donativo – tendo em conta que os donativos podem ser de vários tipos (req. 13), optou-se por categorizar os donativos com a presente entidade.
- Atividade – o conhecimento e agrupamento dos voluntários/doadores por ramo de atividade é uma informação útil e responde aos requisitos req. 5 e req. 11.

A Tabela 1 resume a descrição das entidades identificadas.

Tabela 1 – Descrição das entidades identificadas.

Nome da Entidade	**Descrição**	**Alias**	**Participação**
Funcionário	Termo genérico para descrição de todos os funcionários da Habitat.	Funcionario	Os funcionários trabalham numa comissão e registam os projetos, famílias, doadores, voluntários, equipas e eventos. São também responsáveis por realizar relatórios.
Projeto ou Obra	Termo genérico para descrição de todos os projetos e empreitadas realizadas pela Habitat.	Projeto	Cada projeto é realizado para uma dada família. Ele resulta sempre de uma candidatura aprovada.
Tarefa	Parte de um projeto.	Tarefa	Um projeto tem várias tarefas.
Doador ou Voluntário	O termo Individuo resume todos os doadores e voluntários que contribuem para a Habitat.	Individuo	Pode colaborar em tarefas, fazer donativos e participar em equipas.
Atividade	Termo genérico relativo aos ramos de atividade, como por exemplo construção civil.	Atividade	Todos os indivíduos têm um ramo de atividade associado.
Equipa	Conjunto de voluntários que	Equipa	Uma equipa realiza tarefas.

Nome da Entidade	Descrição	Alias	Participação
	trabalham em conjunto numa dada tarefa.		
Donativo	Termo genérico para descrição de todos os donativos feitos para a Habitat.	Donativo	Os donativos são feitos para os projetos pelos doadores (indivíduos).
Tipo de Donativo	Termo genérico para catalogar os donativos, por exemplo dinheiro, materiais, serviços, ou outros.	TipoDonativo	Todos os donativos têm um tipo.
Evento	Termo para descrever um evento de angariação de fundos.	Evento	Num evento participam vários doadores e são feitos vários donativos.
Família	Termo genérico para descrever um agregado familiar, composto por um representante e outros elementos.	Familia	Uma família candidata-se para a realização de um projeto.
Candidatura	Processo de candidatura de uma família a um projeto de construção da Habitat.	Candidatura	Uma família pode fazer várias candidaturas, podendo uma delas ser aprovada e originar um projeto de construção.
Questão	Termo genérico para descrever as questões de um formulário de candidatura.	Questao	Uma candidatura tem várias questões.
Relatório	Documento de acompanhamento das famílias, após a entrega da nova habitação.	Relatorio	Uma família pode ser alvo de vários relatórios, que são realizados pelos funcionários da Habitat.

4.2. Identificação de Relacionamentos

Assim como a identificação de substantivos nos requisitos foi útil para as entidades, também a identificação de verbos foi útil para estabelecer os relacionamentos.

Os relacionamentos com a entidade “Funcionario” foram os de identificação mais imediata. Os funcionários são responsáveis por criar, entre outros, os registos dos voluntários, famílias, doadores e projetos. Tipicamente este relacionamento tem uma cardinalidade de um para muitos (1..N). Os relacionamentos deste tipo foram designados com o verbo “cria”. A entidade “Funcionario” também se relaciona com a entidade “Relatorio”, uma vez que cada funcionário “faz”, ou pode fazer, relatórios (0..N).

A entidade “Familia” relaciona-se com as entidades “Relatorio” e “Candidatura”. No primeiro, verifica-se que uma família pode ter um ou vários relatórios (0..N). De acordo com o requisito 23, estabeleceu-se o relacionamento “Familia tem Candidatura”, com cardinalidade 1..N, ou seja, uma família pode ter mais do que uma candidatura.
Com a definição da entidade “Questao”, explicada no ponto de identificação das entidades, criou-se o relacionamento entre “Questao” e “Candidatura”. Para este relacionamento,

caracterizado pelo verbo “tem”, definiu-se uma cardinalidade de muitos para muitos (N..N), tipificando assim as questões e associando-as a mais do que uma candidatura.

Ainda de acordo com o requisito 23 definiu-se o relacionamento “aprova”, entre “Candidatura” e “Projeto”. Este relacionamento não é obrigatório, uma vez que podem haver candidaturas não aprovadas, e tem cardinalidade de um para um (1..1). Assim, um projeto só pode ser originado por uma única candidatura.

A entidade “Projeto” é central no presente caso de estudo, concorrendo nesta seis relacionamentos distintos. O primeiro e segundos relacionamentos foram já descritos anteriormente (“Candidatura aprova Projeto” e “Funcionario cria Projeto”).

O terceiro relacionamento resulta do requisito 14 e estabelece que “Projeto recebe Donativo”. Este relacionamento também não é obrigatório, uma vez que poderão não existir donativos para um dado projeto. Tem uma cardinalidade N..N, ou seja, um donativo pode ser usado em vários projetos e um projeto pode receber vários donativos.

O quarto relacionamento é estabelecido com a entidade “Tarefa” e é de carácter obrigatório. Um projeto deve ser dividido em várias tarefas. Tendo em conta que se optou por tipificar as tarefas, o relacionamento “Projeto executa Tarefa” tem uma cardinalidade de muitos para muitos (N..N).

O quinto e sexto relacionamento são ternários e muito semelhantes. São caracterizados pelo verbo “trabalha” e envolvem as entidades “Individuo” (ou “Equipa”), “Tarefa” e “Projeto”. Todas as entidades têm cardinalidade zero ou muitos. Estes relacionamentos surgem devido aos requisitos 7 e 8, que estabelecem que devem ser registadas as tarefas realizadas pelos voluntários/equipas, num determinado projeto.

Tendo em conta que os voluntários podem estar associados a equipas, definiu-se o relacionamento opcional “Individuo tem Equipa”, com cardinalidade muitos para muitos (N..N).

Relativamente aos donativos, foi definido um relacionamento ternário entre “Donativo”, “Evento” e “Individuo”. Este relacionamento permite identificar que doador realizou o donativo, bem como qual foi o evento, a existir, que esteve na sua origem. As entidades “Individuo” e “Evento” têm cardinalidade um ou zero e “Donativo” tem cardinalidade zero ou muitos, uma vez que um doador pode fazer vários donativos, ou um evento dar origem a vários donativos.

Os donativos foram tipificados através do relacionamento “Donativo tem TipoDonativo”, caracterizado por uma cardinalidade de muitos para um.

Os relacionamentos identificados no modelo encontram-se descritos na Tabela 2.

Tabela 2 - Descrição dos relacionamentos identificados.

Entidade	#	Relacionamento	#	Entidade
Funcionario	1..1	cria	0..N	Equipa
Funcionario	1..1	cria	0..N	Individuo
Funcionario	1..1	cria	0..N	Evento
Funcionario	1..1	cria	0..N	Familia
Funcionario	1..1	cria	0..N	Projeto
Funcionario	0..1	faz	0..N	Relatorio
Familia	1..1	faz	1..N	Candidatura
Familia	0..1	tem	0..N	Relatorio
Candidatura	1..N	tem	1..N	Questao
Candidatura	0..1	aprova	0..1	Projeto
Projeto	1..N	executa	1..N	Tarefa
Projeto	0..N	recebe	0..N	Donativo
Donativo	0..N	tem	0..1	TipoDonativo
Individuo	0..N	trabalha	0..N 0..N	Projeto Tarefa
Equipa	0..N	trabalha	0..N 0..N	Projeto Tarefa
Individuo	0..N	tem	0..N	Equipa
Individuo	0..N	tem	0..N	Atividade
Individuo Evento	0..1 0..1	faz	0..N	Donativo

4.3. Identificação de Atributos

No presente capítulo faz-se somente referência aos principais atributos identificados. No entanto, todos os atributos são descritos no dicionário de dados em anexo.

As entidades “Funcionario”, “Familia” e “Individuo” têm um conjunto de atributos que resultam do preenchimento dos formulários respetivos. Dentro deste tipo de atributos são de destacar por exemplo o nome, data de nascimento, NIF, morada, telefone, entre outros. Na “Familia”, este dados dizem respeito ao representante da família.

Na entidade “Funcionario” definiu-se o atributo “tipoFunc”, com o objetivo de identificar a função/comissão a que pertence. Criaram-se também os atributos “username” e “password” para guardar a informação das credenciais de acesso à aplicação.

Na “Familia”, definiu-se um atributo “apelido”, que apesar de não ser único, é útil para identificar a família. Definiu-se também um atributo multi-valor composto para descrever os vários elementos da família (“elementoFam”, ver req. 22). Para além disso, cada projeto

concluído tem associada uma prestação a pagar pela família. Este atributo poderia ter sido associado à entidade “Projeto”, no entanto e por uma questão de simplicidade na aplicação, optou-se por o associar à família. Para descrever esta prestação foi utilizado um atributo multi-valor composto, com uma data e valor associados. Esta definição do atributo “prestacao” permite responder ao requisito 27.

Associado à “Candidatura” foram definidos os atributos “estado” e “rendimento”. O primeiro atributo define o estado atual da candidatura (ex: aprovado), o segundo é o rendimento bruto anual do agregado familiar. Este último atributo é importante para a apreciação da candidatura (ver req. 24).

Um “Projeto” deve ter uma estimativa orçamental (“orcamento”), uma estimativa de prazo (“prazo”), uma data de início (“dataInicio”) e uma data de fecho, após a sua conclusão (“dataFim”).

A entidade “Donativo” tem, para além de uma descrição (“descricao”), uma quantidade inicial (“quantInicial”) e uma quantidade atual (“quantAtual”). Este último atributo é um atributo derivado, que é atualizado sempre que parte do donativo é alocada a um projeto.

A entidade “TipoDonativo” tem uma designação (“designacao”), que pode ser por exemplo dinheiro, cimento, tijolos, serviço. Tem também uma “unidade” que a caracteriza, por exemplo e em correspondência com o anteriormente referido: euros, kg, unidades, N/A.

A entidade “Individuo”, para além dos atributos de descrição geral (nome, telefone, NIF, etc.), tem um conjunto de quatro atributos booleanos: “isColectivo”, “isParceiro”, “isDoador” e “isVoluntario”. Os últimos dois servem para identificar se o individuo se trata de um doador e/ou voluntário. Os dois primeiros servem para identificar se se trata de uma pessoa colectiva e de um parceiro da Habitat, respetivamente.

Relativamente aos atributos dos relacionamentos, foi definido em todos os relacionamentos do tipo “cria” com a entidade “Funcionario” uma data de criação (por exemplo “dataCriaInd” para o relacionamento com “Individuo”).

No relacionamento entre “Projeto” e “Tarefa” foram definidos quatro atributos de datas: uma data de início e fim do planeamento de uma tarefa (“dataPlInicio” e “dataPlFim”); de forma análoga, foi definida uma data de início e fim de uma tarefa executada (“dataExInicio” e “dataExFim”). Com a definição destas quatro por cada tarefa, é possível realizar a comparação do executado com o planeado no início da obra (ver req. 19).

Os relacionamentos ternários entre “Equipa”/”Individuo”, “Projeto” e “Tarefa”, têm uma data (“dataTrabalho”) e duração (“duracao”) associadas, de forma a permitir o registo das tarefas.

No relacionamento ternário entre “Donativo”, “Evento” e “Individuo”, foi definida também uma data para registar o momento do donativo (“dataDon”).

No relacionamento entre “Donativo” e “Projeto” foram definidos dois atributos: uma data em que o donativo foi alocado ao projeto (“dataDonProj”) e a quantidade alocada (“quantDonProj”).

Os restantes atributos são facilmente identificados pela consulta do dicionário de dados e diagrama do modelo (elementos em anexo).

4.4. Definição de Domínios de Atributos

A definição do domínio de cada atributo pode ser consultada no dicionário de dados anexo. No presente capítulo, apresenta-se somente uma descrição dos critérios gerais para a sua definição.

Seguidamente apresentam-se alguns critérios gerais adotados para a definição dos domínios:

- Todas as datas foram definidas com o tipo de dados “DATE”;
- Os números de telefone e telemóvel foram definidos com o tipo VARCHAR(20), de forma a puderem incluir caracteres não numéricos;
- O código postal foi definido com o tipo VARCHAR(10), de forma a puder incluir caracteres não numéricos;
- Os nomes e moradas foram definidos com o tipo VARCHAR(75);
- As descrições e designações foram definidas um número maior de caracteres, VARCHAR(100) e VARCHAR(200);
- Para os ficheiros a anexar à BD foi definido o tipo BLOB;
- As chaves foram todas definidas com o tipo INT.
- As questões e atributos booleanos foram definidos com o tipo BOOLEAN.
- Para o atributo “estado”, em “Candidatura”, foi definido um tipo ENUM, com o seguinte domínio: ‘APROVADO’, ‘NAOAPROVADO’, ‘NAOACEITE’ e ‘PENDENTE’.
- Também para o tipo de funcionário (“tipoFunc”) foi definido o tipo ENUM, com o seguinte domínio: ‘ADMIN’, ‘FAM’, ‘FUNDOS’, ‘OBRAS’.

Para o atributo “quantAtual”, que descreve a quantidade atual de um donativo, definiu-se um domínio positivo (INT UNSIGNED), uma vez que este atributo não admite valores negativos.

4.5. Identificação de Chaves Primárias e Candidatas

Para todas as entidades foi definido um atributo de identificação, que serviu de chave primária. A estes atributos foi associado o prefixo “id”, como por exemplo “idFunc”, “idFam” ou “idCand”.

No caso das entidades “Individuo” e “Funcionario”, o atributo NIF pode servir também para uma identificação unívoca, sendo por isso uma chave candidata.

4.6. Verificação de Redundância

Os relacionamentos de um para um são uma das fontes de redundância, no entanto no presente modelo só existe um relacionamento desse tipo, entre “Candidatura” e “Projeto”. Estas entidades têm finalidades distintas e bem definidas, pelo que não existe redundância a este nível.

Foram também analisados possíveis caminhos diferentes entre entidades, de forma a avaliar relacionamentos redundantes. A esse nível pode-se verificar que a “Equipa”/”Individuo” estão relacionados com o “Projeto” e “Tarefa”. Esta última também se encontra relacionada com o “Projeto”, no entanto os relacionamentos representam situações diferentes, pelo que se entende que não existe redundância. Também no caso da “Equipa” e “Individuo” existem dois relacionamentos possíveis, um através da “Tarefa” e outro direto entre elas. Verifica-se que não existe redundância, uma vez que quando uma “Equipa” “trabalha” na “Tarefa”, o “Individuo” que pertence à equipa não se relaciona com a tarefa diretamente, por uma questão de registo separado dos voluntários e equipas.

Após as verificações de redundância fundiu-se a entidade “Voluntario” com a entidade “Doador” por poderem ser representados por uma só “Individuo”, uma vez que têm uma caracterização semelhante. Poderão existir funcionários que são simultaneamente doadores ou voluntários, no entanto, entende-se que esta possível redundância é aceitável.

4.7. Validação com Transações do Utilizador

A validação do modelo com transações do utilizador foi realizada de acordo com as *queries* enunciadas no capítulo 2 - Levantamento e Análise de Requisitos. Foi utilizada uma abordagem descritiva em detrimento de uma descrição gráfica, com os caminhos das transações descritos no diagrama do modelo.

Transação 1: Listar os projetos para os quais um doador contribui.

Os detalhes dos projetos encontram-se definidos na entidade “Projeto” e os detalhes dos doadores na entidade “Individuo”. Para filtrar os indivíduos que são doadores é necessário verificar que o atributo “isDoador” tem valor verdadeiro. Utilizando o relacionamento “Individuo faz Donativo” seguido do relacionamento “Projeto recebe Donativo” é possível fazer a correspondência entre o doador e o projeto, listando o pretendido. Retendo os atributos do segundo relacionamento é possível conhecer a data e quantidade em que o donativo foi alocado.

Transação 2: Listar os projetos para os quais um voluntário trabalhou individualmente.

Os detalhes dos projetos e voluntários encontram-se definidos nas entidades “Projeto” e “Individuo”, respetivamente. De forma a filtrar os voluntários é necessário verificar que o atributo “isVoluntario” tem valor verdadeiro. Utilizando o relacionamento “Individuo trabalha Projeto” é possível fazer a correspondência e apresentar a listagem requerida. Retendo os atributos do relacionamento é possível conhecer também as datas e durações dos trabalhos realizados. Adicionando a terceira entidade do relacionamento (“Tarefa”) é também possível conhecer quais as tarefas que foram realizadas.

Transação 3: Listar os donativos alocados a um determinado projeto.

Os detalhes dos projetos e donativos encontram-se definidos nas entidades “Projeto” e “Donativo”, respetivamente. O relacionamento “Projeto recebe Donativo” permite fazer a listagem pretendida, nomeadamente através dos atributos do relacionamento (“dataDonProj”, “quantDonProj”) que descriminam a quantidade alocada do donativo e a data em que foi alocado.

Transação 4: Identificar a família a que corresponde um determinado projeto.

Os detalhes dos projetos e famílias encontram-se definidos nas entidades “Projeto” e “Familia”, respetivamente. Seguindo pela ordem descrita os relacionamento “Familia faz Candidatura” e “Projeto aprova Candidatura” é possível fazer a correspondência entre uma família e um projeto.

Transação 5: Listar os relatórios realizados para uma dada família.

Os detalhes dos projetos e relatórios encontram-se definidos nas entidades “Projeto” e “Relatorio”, respetivamente. O relacionamento “Familia tem Relatorio” permite apresentar a listagem requerida.

Transação 6: Listar o número de doadores por atividade.

Os detalhes dos doadores e atividades encontram-se definidos nas entidades “Individuo” e “Atividade”, respetivamente. Para filtrar os indivíduos que são doadores é necessário verificar

que o atributo “isDoador” tem valor verdadeiro. O relacionamento “Individuo tem Atividade” permite fazer a correspondência entre doadores e atividades. Após realizada a listagem é necessário agrupar os doadores por atividade e fazer a respetiva contagem.

4.8. Validação do Modelo com o Utilizador

O modelo representado em anexo foi validado em reunião havida com o Prof. Orlando Belo no dia 21/11/2014.

Não foi possível realizar uma validação direta com a Habitat.

5. Modelo Lógico

No presente capítulo descrevem-se as etapas de conversão do modelo conceptual para o modelo lógico e a validação do mesmo.

O diagrama foi desenvolvido em Mysql Workbench, que utiliza a notação “Crow’s Feet”, e pode ser consultado no Anexo II.

Por uma questão de organização do diagrama, dividiu-se o modelo lógico em quatro *layers*, um *layer* por comissão (famílias, fundos e construção) e outro para tabelas mais relacionadas com a administração.

5.1. Conversão do Modelo Conceptual

Nos pontos seguintes descrevem-se os esquemas das relações criadas, apresentando a descrição do processo em seis etapas: entidades fortes,

Os esquemas das tabelas foram representados de acordo com o seguinte exemplo:
Tabela (atributo1, atributo2, atributo3)
Primary Key atributo1
Foreign Key atributo2 **references** outraTabela(atributo2)

Por uma questão de simplificação da representação encontram-se omissos alguns atributos secundários.

5.1.1 Entidades fortes

As entidades fortes seguintes são simplesmente as entidades do modelo conceptual que não têm uma chave estrangeira como chave primária convertidas em tabelas do modelo lógico.

Funcionario (idFunc, tipoFunc, nome, nif, morada, localidade, codigoPostal, dataNascimento, telemovel, telefone)
Primary Key idFunc

Relatorio (idRel, dataRel, obsRel, ficheiroRel)
Primary Key idRel

Questao (idQuestao, descricao)
Primary Key idQuestao

Equipa (idEq, designacao, nacionalidadeEq)
Primary Key idEq

Familia (idFam, apelido, nome, morada, telefone, dataNascimento, nif)
Primary Key idFam

Candidatura (idCand, estado, rendimento)
Primary Key idCand

Projeto (idProj, descricao, prazo, dataInicio, dataFim)
Primary Key idProj

Tarefa (idTar, designacao)
Primary Key idTar

TipoDonativo (idTipoDon, unidade, designacao)
PrimaryKey idTipoDon

Donativo (idDon, quantInicial, descricao, quantAtual)
Primary Key idDon

Evento (idEv, designacao, data)
Primary Key idEv

Individuo (idIndiv, nif, nacionalidadeIndiv, nome, dataNascimento, profissao, morada, codigoPostal, localidade, telemovel)
Primary Key idIndiv

Atividade (idAtividade, designacao)
Primary Key idAtividade

5.1.2 Relacionamentos de 1 para muitos (1:n)

Como a entidade “Relatorio” tem dois relacionamentos de 1 para muitos com as entidades “Familia” e “Funcionario”, é necessário adicionar dois atributos (“idFam” e “idFunc”) à tabela, que representam as chaves estrangeiras de “Familia” e “Funcionario”.

Relatorio (idRel, dataRel, obsRel, ficheiroRel, idFam, idFunc)
Primary Key idRel
Foreign Key idFam **references** Familia(idFam),
idFunc **references** Funcionario(idFunc)

Na tabela “Equipa” é necessário adicionar dois novos atributos aos anteriormente expostos. Tal se deve ao relacionamento de um para muitos com a entidade “Funcionario”, é necessário adicionar uma chave estrangeira (“idFunc”) e o atributo do relacionamento (“dataCriaEq”).

Equipa (idEq, designacao, nacionalidadeEq, dataCriaEq, idFunc)
Primary Key idEq
Foreign Key idFunc **references** Funcionario(idFunc)

Na tabela “Familia”, anteriormente referida, foram adicionados dois novos atributos devido ao relacionamento de um para muitos com a entidade “Funcionario”. Ou seja, uma chave estrangeira vinda de “Funcionario” (“idFunc”) e um atributo do relacionamento (“dataCriaFam”).

Familia (idFam, apelido, nome, morada, telefone, dataNascimento, nif, dataCriaFam,idFunc)
Primary Key idFam
Foreign Key idFunc **references** Funcionario(idFunc)

Na tabela “Candidatura”, anteriormente referida, foram adicionados dois novos atributos devido ao relacionamento de um para muitos com a entidade “Familia”: uma chave estrangeira vinda de “Familia” (“idFam”) e um atributo do relacionamento “dataCan”.

Candidatura (idCand, estado, rendimento, dataCan, idFam)
Primary Key idCand
Foreign Key idFam **references** Familia(idFam)

À tabela “Projeto” foram adicionados dois novos atributos devido ao relacionamento de um para muitos com “Funcionario”: uma chave estrangeira de “Funcionario” (“idFunc”) e um atributo do relacionamento (“dataCriaProj”).

Projeto (idProj, descricao, prazo, dataInicio, dataFim, dataCriaProj, idFunc)
Primary Key idProj

Foreign Key idFunc **references** Funcionario(idFunc)

Na tabela “Donativo” foram adicionados dois novos atributos, que são chaves estrangeiras e resultam dos relacionamentos de 1 para muitos com “Funcionario” e “TipoDonativo”.

Donativo (idDon, quantInicial, descricao, quantAtual, idFunc, idTipoDon)
Primary Key idDon
Foreign Key idFunc **references** Funcionario(idFunc),
idTipoDon **references** TipoDonativo(idTipoDon)

Na tabela “Evento” foram adicionados dois novos atributos devido ao relacionamento de um para muitos com “Funcionario”: uma chave estrangeira de “Funcionario” (“idFunc”) e um atributo do relacionamento “dataCriaEv”.

Evento (idEv, designacao, data, dataCriaEv, idFunc)
Primary Key idEv
Foreign Key idFunc **references** Funcionario(idFunc)

Na tabela “Individuo” foram adicionados dois novos atributos devido ao relacionamento de um para muitos com “Funcionario”: uma chave estrangeira de “Funcionario” (“idFunc”) e um atributo do relacionamento “dataCriaInd”.

Individuo (idIndiv, nif, nacionalidadeIndiv, nome, dataNascimento, profissao, morada,
codigoPostal, localidade, telemovel, dataCriaInd,idFunc)
Primary Key idIndiv
Foreign Key idFunc **references** Funcionario(idFunc)

5.1.3 Relacionamentos de 1 para 1 (1:1)

O relacionamento de um para um entre “Projeto” e “Candidatura” é um relacionamento de participação obrigatória do lado do projeto, logo foi adicionado um novo atributo à tabela “Projeto”, que é uma chave estrangeira de “Candidatura” (“idCan”).

Projeto (idProj, descricao, prazo, dataInicio, dataFim, idFunc, idCan)
Primary Key idProj
Foreign Key idFunc **references** Funcionario(idFunc),
idCan **references** Candidatura(idCand)

5.1.4 Relacionamentos de muitos para muitos (n:m)

Devido ao relacionamento de muitos para muitos entre “Candidatura” e “Questao” criou-se uma nova tabela designada “CandidaturaQuestao”. Esta tabela contém três atributos: duas chaves estrangeiras que são simultaneamente chaves primárias, “idCand” e “idQuestao”, e um atributo resultante do relacionamento (“resposta”).

CandidaturaQuestao (idCand, idQuestao, resposta)
Primary Key idCand, idQuestao
Foreign Key idCand **references** Candidatura(idCand),
idQuestao **references** Questao(idQuestao)

Devido ao relacionamento de muitos para muitos entre “Tarefa” e “Projeto” criou-se uma nova tabela designada “TarefaProjeto”, que contém dois atributos que são chaves estrangeiras, “idTar” e “idProj”, e quatro atributos relativos ao relacionamento, “dataPlInicio”, “dataPlFim”, “dataExInicio” e “dataExFim”.

TarefaProjeto (idProj, idTar, dataPlInicio, dataPlFim, dataExInicio, dataExFim)
Primary Key idProje, idTar
Foreign Key idProj **references** Projeto(idProj),
idTar **references** Tarefa(idTar)

Devido ao relacionamento de muitos para muitos entre “Equipa” e “Individuo” criou-se uma nova tabela designada “EquipaIndividuo”, que contém dois atributos que são chaves estrangeiras, “idIndiv” e “idEq”.

EquipaIndividuo (idIndiv, idEq)
Primary Key idIndiv, idEq
Foreign Key idIndiv **references** Individuo(idIndiv),
idEq **references** Equipa(idEq)

Devido ao relacionamento de muitos para muitos entre “Donativo” e “Projeto” criou-se uma nova tabela designada “DonativoProjeto”, que contém dois atributos que são chaves estrangeiras, “idDon” e “idProj”, e dois atributos do relacionamento, “dataDonProj” e “quantDonProj”.

DonativoProjeto (idDon, idProj, dataDonProj, quantDonProj)
Primary Key idDon, idProj
Foreign Key idDon **references** Donativo(idDon),
isProj **references** Projeto(idProj)

Devido ao relacionamento de muitos para muitos entre “Individuo” e “Atividade” criou-se uma nova tabela designada “IndividuoAtividade”, que contém dois atributos que são chaves estrangeiras, “idIndiv” e “idAtividade”.

IndividuoAtividade (idIndiv, idAtividade)
Primary Key idIndiv, idAtividade
Foreign Key idIndiv **references** Individuo(idIndiv),
idAtividade **references** Atividade(idAtividade)

5.1.5 Relacionamentos complexos

O relacionamento ternário entre “Equipa”, “Tarefa” e “Projeto” trata-se de um relacionamento de muitos para muitos, em todas as entidades que participam. Assim, criou-se uma nova tabela designada “EquipaTarefaProjeto”, que tem como chave primária uma chave composta pelas chaves primárias das outras três tabelas (“idEq”, “idProj” e “idTar”) e recebe também os atributos do relacionamento referido (“dataTrabalho” e “duracao”).

EquipaTarefaProjeto (idEq, idProj, idTar, dataTrabalho, duracao)
Primary Key idEq, idProj, idTar
Foreign Key idEq **references** Equipa(idEq),
idProj **references** TarefaProjeto(idProj),
idTar **references** TarefaProjeto(idTar)

O relacionamento ternário entre “Individuo”, “Tarefa” e “Projeto” é muito semelhante ao anteriormente descrito, bastando para tal substituir a entidade “Equipa” por “Individuo”. Assim, criou-se uma nova tabela “IndividuoTarefaProjeto”, tal como seguidamente descrito.

IndividuoTarefaProjeto (idIndiv, idProj, idTar, dataTrabalho, duracao)
Primary Key idIndiv, idProj, idTar
Foreign Key idIndiv**references** Individuo(idIndiv),
idProj **references** TarefaProjeto(idProj),
idTar **references** TarefaProjeto(idTar)

O relacionamento ternário entre “Donativo”, “Evento” e “Individuo” é de um para um para muitos, respetivamente. Assim, não foi necessário criar uma nova tabela para acomodar este relacionamento. No entanto, foi necessário colocar as chaves estrangeiras de “Evento” e “Individuo” na tabela “Donativo”. Foi também adicionado o atributo do relacionamento “dataDon”.

Donativo (idDon, quantInicial, descricao, quantAtual, idFunc, idTipoDon, dataDon, idEv, idIndiv)
Primary Key idDon
Foreign Key idFunc **references** Funcionario(idFunc),
idTipoDon **references** TipoDonativo(idTipoDon),
idEv **references** Evento(idEv),
idIndiv **references** Individuo(idIndiv)

5.1.6 Atributos multi-valor

Para o atributo multi-valor "ElementoFamilia", da entidade "Familia", foi criada uma tabela nova, com uma chave estrangeira de "Familia", assim como todos os atributos compostos de "ElementoFamilia" ("nome", "parentesco", "dataNascimento", etc.). Foi definida uma chave primária para identificação unívoca de cada elemento da família ("numElemento").

ElementoFamilia (numElemento, idFamilia, nome, parentesco, dataNascimento, estadoCivil, ocupacao, escolaridade)
Primary Key numElemento
Foreign Key idFamilia **references** Familia(idFam)

Para o atributo multi-valor "Prestacao", da entidade "Familia", foi criada uma tabela nova, com uma chave estrangeira de "Familia" e os atributos compostos de "Prestacao" ("valor", "data"). Foi definida uma chave primária para identificação unívoca de cada prestação ("idPrestacao"), bem como uma chave estrangeira do funcionário que criou a prestação ("idFunc").

Prestacao (idPrestacao, idProj, idFunc, valor, data)
Primary Key idPrestacao
Foreign Key idFamilia **references** Familia(idFam),
idFunc **references** Funcionario(idFunc)

5.2. Validação através da Normalização

De acordo com a metodologia seguida, a base de dados está garantidamente normalizada na terceira forma normal.

5.3. Validação com Transações do Utilizador

No presente ponto são validadas as *queries* descritas no capítulo 2 - Levantamento e Análise de Requisitos. O procedimento de demonstração semelhante ao já realizado para o modelo conceptual, utilizando neste caso as tabelas anteriormente descritas. As descrições são complementadas por árvores de álgebra relacional, com a descrição matemática das operações.

Transação 1: Listar os projetos para os quais um doador contribui.
Com a ajuda da tabela “DonativoProjeto” pode-se verificar, entre todos os donativos feitos, os que foram usados nos projetos pretendidos. Juntando os dados de “DonativoProjeto” com a tabela “Projeto” e “Donativos” pode-se concluir onde foram usados os donativos de um doador.

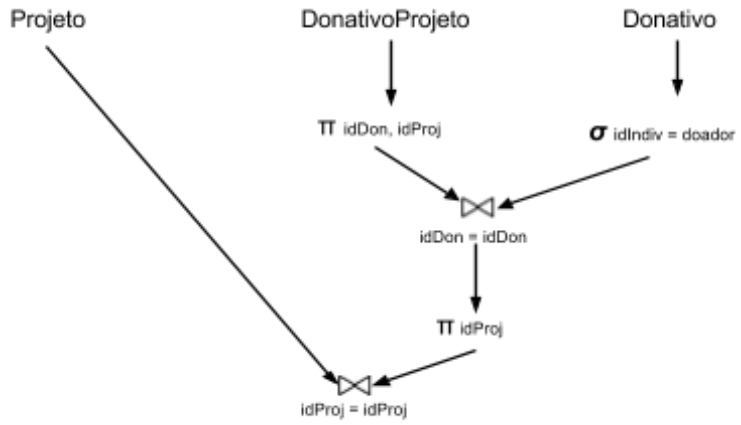


Figura 3 - Transação 1, árvore de álgebra relacional.

Transação 2: Listar os projetos para os quais um voluntário trabalhou individualmente.
Para listar os projetos para os quais um voluntário trabalhou individualmente basta pesquisar na tabela “IndividuoTarefaProjeto” os “ids” dos projetos em que o voluntário trabalhou e depois, através da tabela “Projeto”, retirar a informação dos projetos em que o voluntário trabalhou.

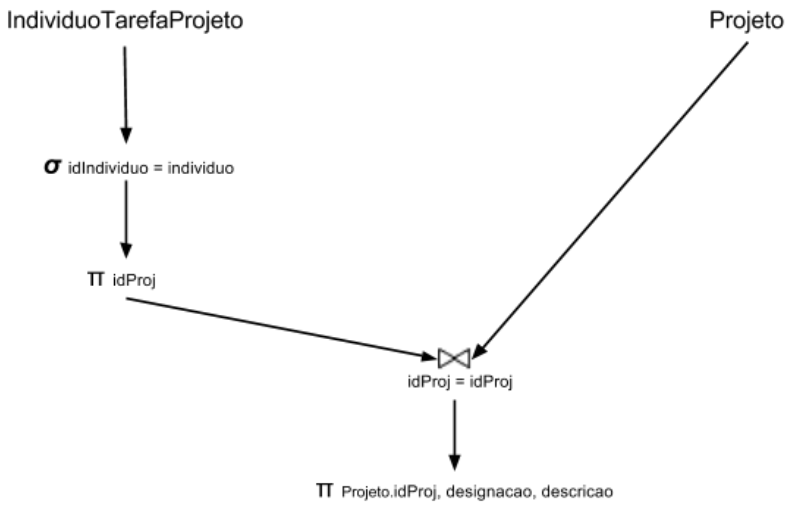


Figura 4 - Transação 2, árvore de álgebra relacional.

Transação 3: Listar os donativos alocados a um determinado projeto.

Para listar os donativos alocados a um determinado projeto é necessário utilizar a tabela "DonativoProjeto", para obter o "id" de todos os donativos usados num determinado projeto. De seguida, utiliza-se a tabela "Donativo" para obter os dados desses donativos.

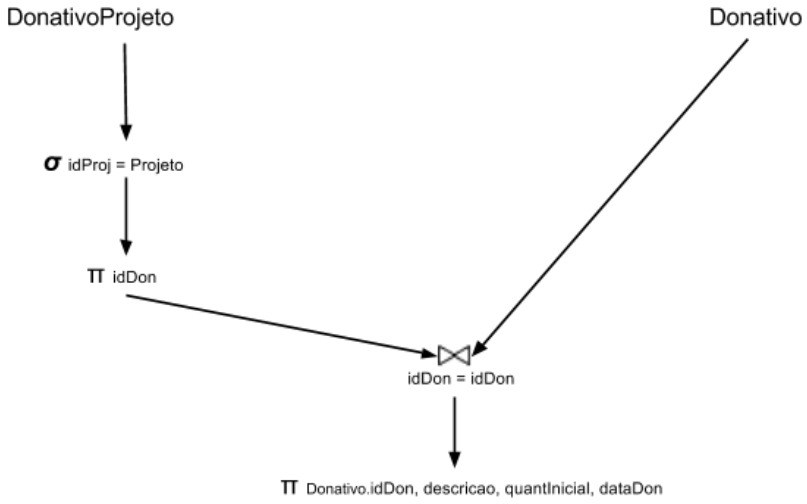


Figura 5 - Transação 3, árvore de álgebra relacional.

Transação 4: Identificar a família a que corresponde um determinado projeto.

Através do campo "idCand" da tabela "Projeto" é possível conhecer qual a candidatura que deu origem ao projeto pretendido. Posteriormente, pesquisa-se na tabela "Candidatura" a família dessa candidatura e por fim consulta-se na tabela "Familia" os dados da família.

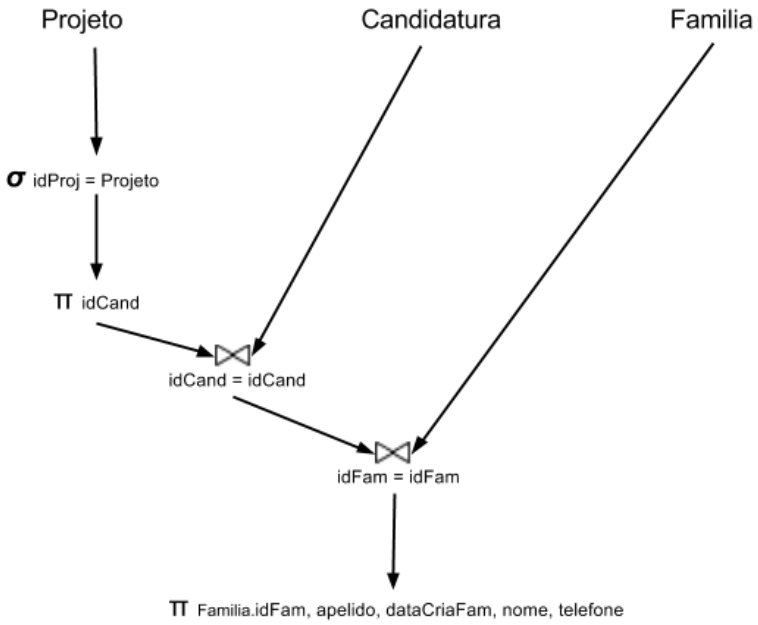


Figura 6 - Transação 4, árvore de álgebra relacional.

Transação 5: Listar os relatórios realizados para uma dada família.

Para listar os relatórios realizados para uma dada família basta utilizar a tabela “Relatorio” e fazer uma pesquisa com o “id” da família.

Figura 7 - Transação 5, árvore de álgebra relacional.

Transação 6: Listar o número de doadores por atividade.

Para listar o número de doadores por atividade é necessário primeiro consultar a tabela “Individuo”, selecionando os indivíduos que são doadores. De seguida, utiliza-se a tabela “IndividuoActividade” para determinar a atividade de cada doador. Juntando a tabela “Actividade” é possível conhecer os detalhes de cada atividade. Por fim efetua-se uma contagem do total por cada atividade.

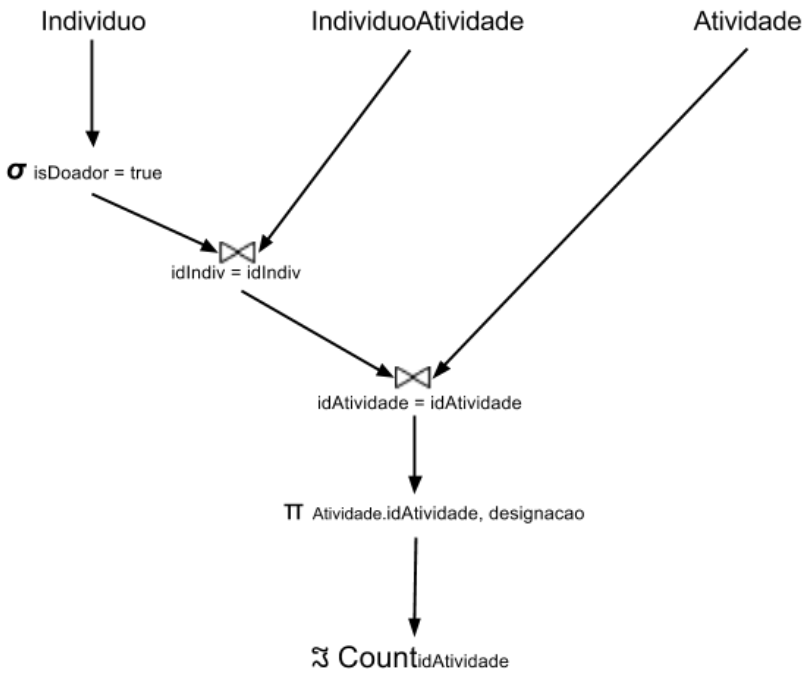


Figura 8 - Transação 6, árvore de álgebra relacional.

5.4. Verificação de Restrições de Integridade

Para verificar as restrições de integridade foram considerados os seguintes tipos de restrições: dados necessários, restrições de domínio dos atributos, multiplicidade, integridade da entidade, integridade referencial e restrições gerais.

Nas restrições de dados necessários, aplicou-se a restrições de valores não nulos (*not null*) aos atributos com valor válido obrigatório, como por exemplo o nome de um doador, ou o valor do donativo.

De forma a garantir a integridade da entidade as chaves primárias das tabelas foram também definidas como não nulas.

Foram também definidas algumas restrições de domínio, como por exemplo no atributo "estado" da tabela "Candidatura" definiu-se um tipo enumerado (ENUM), com o seguinte domínio: aprovado, não aprovado, não aceite pela família ou pendente. O domínio do atributo relativo ao tipo de funcionário também foi restringido às seguintes possibilidades: Administrador, Comissão de Famílias, Comissão de Fundos ou Comissão de Construção (tipo ENUM). Relativamente, ao atributo da quantidade atual do donativo ("quantAtual"), presente na tabela "Donativo", definiu-se um domínio com valores não negativos através do tipo de dados INT UNSIGNED.

A multiplicidade foi assegurada no modelo conceptual e na conversão para o lógico.

A integridade referencial foi garantida com a definição da restrição “not null” nas chaves estrangeiras dos relacionamentos obrigatórios, de que é exemplo a chave estrangeira dos funcionários (“idFunc”), nas tabelas onde é necessário conhecer os funcionários que são responsáveis pela criação do registo. Outro problema com a integridade referencial pode ocorrer quando se insere um valor numa chave estrangeira que não existe na tabela original da chave estrangeira ou quando o registo da tabela original é apagado. Estas questões podem ser resolvidas com restrições que definem a ação a tomar após remoção e atualização de uma chave estrangeira. Essas opções são por exemplo c*ascade*, *set null*, *set default*. No presente caso não foram definidas restrições desta natureza, ou seja, definiu-se o seguinte:

```
ON DELETE NO ACTION
ON UPDATE NO ACTION
```

Não foram identificadas nem definidas restrições gerais aos dados.

5.5. Validação do Modelo com o Utilizador

Não foi possível realizar a validação do modelo lógico.

5.6. Verificação sobre Expansão Futura do Modelo

De uma forma geral, não se antecipa a necessidade de criação de mais tabelas ou relacionamentos. Contudo, de forma a prever uma maior flexibilidade na expansão futura do modelo, foram criadas algumas tabelas com tipificação de questões, tarefas, donativos ou atividades dos voluntários/doadores (tabelas “Atividade”, “Tarefa”, “TipoDonativo” e “Questao”). As tabelas em causa deverão ser mantidas pela administração da BD e permitem, por exemplo, adicionar de uma forma fácil novas tarefas de construção, novas questões relativas aos processos de candidatura, novos tipos de donativos.

Poder-se-ia ter criado uma tabela para tipos de funcionários, mas atendendo à dimensão diminuta da organização, optou-se por simplificadamente considerar os tipos de funcionários através de um atributo enumerado.

6. Modelo Físico

O Motor escolhido para implementar fisicamente esta base de dados foi o MySql, pois é uma solução gratuita – o ideal para uma associação sem fins lucrativos, é fácil de instalar e têm uma grande comunidade, o que facilita a resolução de problemas futuros.

6.1. Implementação e Conversão do Modelo Lógico

A conversão para o modelo logico foi imediata, uma vez que foi utilizada o *software* MySQL workbench. Assim, foi exportada a script de criação da BD (*forward engineering*) SQL, que foi depois executada no mysql server.

O mecanismo de armazenamento usado para todas as tabelas do esquema foi o InnoDB, que é o mecanismo de armazenamento usado por omissão no MySQL Workbench. Suporta chaves estrageiras e transações, entre outras vantagens.

Para a gestão dos atributos derivados foram criados *triggers*, que se encontram descritos no sub-item seguinte.

Foram definidos algumas restrições usuais aos dados, nomeadamente:

- Valores não nulos em chaves e atributos obrigatórios (NOT NULL);
- Restrição do domínio de valores não negativos, como na quantidade atual dos donativos (UNSIGNED);
- Restrições nas chaves candidatas, como o NIF dos voluntários, doadores ou funcionários (UNIQUE).

Relativamente aos índices, foram definidos índices nas chaves primárias, estrangeiras e candidatas de cada tabela.

6.1.1 Triggers – SQL

Foram criados *triggers* para o cálculo da quantidade atual de um Donativo (“quantAtual”), para que os registos sejam atualizados automaticamente quando se insere uma nova entrada em “DonativoProjeto”.

De acordo com o exposto, na tabela “DonativoProjeto” foram definidos três *triggers* (ReporStock, TirarStock, TirarStockUpdate). Estes *triggers* são acionados antes (BEFORE) da realização das operações de inserção, atualização e remoção. Tomou-se esta opção de forma a controlar *a priori* a existência de quantidades atuais negativas. Expõem-se seguidamente os códigos respetivos.

```
CREATE TRIGGER `TirarStock` BEFORE INSERT ON `DonativoProjeto` FOR EACH ROW
BEGIN
    UPDATE Donativo Set quantAtual = quantAtual - NEW.quantDonProj
    WHERE idDon = NEW.idDon;
    IF (@@error_count > 0) then
        SIGNAL SQLSTATE '45000'
            SET MESSAGE_TEXT = 'Quantidade Actual inferior a 0', MYSQL_ERRNO = 1001;
    END IF;
END

CREATE TRIGGER `TirarSockUpdate` BEFORE UPDATE ON `DonativoProjeto` FOR EACH ROW
BEGIN
    UPDATE Donativo Set quantAtual = quantAtual + OLD.quantDonProj - NEW.quantDonProj
    WHERE idDon = NEW.idDon;
        IF (@@error_count > 0) then
        SIGNAL SQLSTATE '45000'
            SET MESSAGE_TEXT = 'Quantidade Actual inferior a 0', MYSQL_ERRNO = 1001;
        END IF;
END

CREATE TRIGGER `ReporStock` BEFORE DELETE ON `DonativoProjeto`
FOR EACH ROW
BEGIN
        UPDATE Donativo Set quantAtual = quantAtual + OLD.quantDonProj
       WHERE idDon = OLD.idDon;
END
```

6.2. Análise de Transações

Foram identificadas algumas das transações que iram existir no dia-a-dia da Habitat. As transações identificadas são resultantes de atributos multi-valor e outras são de atualizações de registos.

As transações identificadas foram:

A- Registo de uma família implica a inserção de registos nas tabelas: “Familia”, “Candidatura”, “CandidaturaQuestao”, “ElementoFamilia”.

B- Aprovação de uma candidatura implica a atualização do estado da candidatura e inserção nas tabelas: “Projeto”, “TarefaProjeto”.

C – Inserção de novo voluntário ou doador implica a inserção de registos nas tabelas “Individuo” e “IndividuoActividade”.

D – Criação de uma equipa implica inserção em “Equipa” e “EquipaIndividuo”.

O mapa de transações completo pode ser encontrado em anexo com todas as transações aqui descritas.

6.3. Implementação de Transações

Devidos às limitações que o mysql oferece relativamente à passagem de argumentos do tipo *array* para os procedimentos. Coisa que era essencial para a passagem de argumentos de, como por exemplo, os dados dos vários elementos de uma determinada família (Transação A). As transações foram criadas a nível aplicacional, com base no seguinte esqueleto de código SQL:

```
START TRANSACTION;
INSERT INTO Familia ( … ) VALUES ( … )
INSERT INTO Candidatura ( … ) VALUES ( … )
INSERT INTO CandidaturaQuestao ( … ) VALUES ( … )
INSERT INTO ElementoFamilia ( … ) VALUES ( … )

COMMIT;
ou
ROLLBACK;
```

O *commit* será executado quando todas as *queries* tiverem sucesso, ou seja sem erros. O *rollback* é executado caso ocorra algum erro durante a transação, anulando a totalidade das operações.

6.4. Definição de Vistas dos Utilizadores

Foi criada apenas uma vista, para facilitar a leitura dos dados da tabela “TarefaProjeto”. Esta vista inclui para além do “idTar” a designação da tarefa. Desta forma, evitou-se a realização da junção sistemática entre as duas tabelas.

```
CREATE VIEW `ViewTarefaProjeto` AS
Select  idProj, t.idTar as idTar, t.designacao as designacaoTar,
        dataPlInicio, dataPlFim, DataExInicio, DataExFim
FROM TarefaProjeto tp inner join Tarefa t
        ON t.idtar = tp.idtar;
```

6.5. Políticas de Segurança

As políticas de segurança que foram tomadas são: criação de perfis de utilização e backups dos dados. Estes dois tópicos serão descritos nos pontos seguintes.

6.5.1 Acesso

Definiram-se seis perfis de utilização, a saber: Administrador, Funcionário de Comissão de Obras, Funcionário de Comissão de Fundos, Funcionário de Comissão Famílias, Convidado e utilizador para efetuar logins.

O administrador terá acesso total a todas as operações e tabelas.

O funcionário da comissão de famílias terá acesso de inserção, remoção e edição às tabelas de “Familia”, “ElementoFamilia”, “Relatorio”, “Candidatura”, “CandidaturaQuestao”, “Prestacao”.

O funcionário da comissão de Obras terá acesso de inserção, remoção e edição às tabelas de “Projeto”, “TarefaProjeto”, “IndividuoTarefaProjeto”, “EquipaTarefaProjeto”, “DonativoProjeto”.

O funcionário da comissão de Fundos terá acesso de inserção, remoção e edição às tabelas de “Donativo”, “Evento”, “Individuo”, “EquipaIndividuo”, “IndividuoActividade”, “Equipa”.

O Convidado terá acesso de consulta a todas as tabelas das comissões.

E por final um utilizador de logins que terá apenas acesso às colunas “password” e “username” na tabela “Funcionario”. Este utilizador deverá apenas ser usado pela aplicação.

```
CREATE USER 'loginCheck' IDENTIFIED BY <password>;
GRANT SELECT (username, password) ON TABLE `habitat`.`Funcionario` TO 'loginCheck';
```

Todos os perfis de utilização têm acesso de consulta a toda a informação.

6.5.2 Dados

Uma vez que a habitat não é uma organização de grandes dimensões, e o crescimento dos dados não é significado. É suficiente fazer *backup* de todos os dados mensalmente. Ou seja, idealmente todos os meses a habitat irá ter pelo menos dois *backups* de todos os dados dos últimos dois meses.

Estes *backups* podem ser guardados num disco externo, no caso de avaria das máquinas, por exemplo devido a um pico de corrente.

O custo deste plano de *backups* será reduzido quer em termos de tempo como espaço de disco.

Comandos a utilizar para o *backup* mensal:

```
mysqldump -uroot -p habitat > backup.sql
```

A política a aplicar deverá ser mais bem discutida com a Habitat. Caso esta se revele insuficiente, pode ser estudada uma nova política com *backups* incrementais.

6.6. Povoamento da BD

De forma a testar a BD criada, foi realizado um povoamento de reduzida dimensão em todas as tabelas.

O povoamento realizado serviu também de base para a estimativa do espaço de disco necessário, bem como para teste das *queries* descritas no capítulo dos requisitos.

Nas tabelas seguintes do relatório pode ser consultado o povoamento realizado e que serviu de base às *queries* do capítulo 6.8.

A *script* de povoamento de todas as tabelas pode ser consultada no Anexo VI.

Tabela 3 – Povoamento da tabela “Funcionario” (foram omitidos alguns atributos).

idFunc	tipoFunc	Nome	dataNascimento	username	password
1	ADMIN	Rui Oliveira	1994-11-10	User	Password
2	FAM	Rufino Felipe	1991-02-23	Rufino	Indecifravel
3	FUNDOS	Teresa Frederica	1993-02-02	Teresa	Teresa
4	OBRAS	Antonio Mafalda	1993-09-15	Antonio	tone

Tabela 4 - Povoamento da tabela “Tarefa”.

idTar	designacao
1	Colar Papel Interior
2	Pintar exterior
3	Colocar soalho
4	Instalaçao de tubagens
5	Limpar exterior
6	Instalação de pavimento
7	Instalar quadro eléctrico

Tabela 5 - Povoamento da tabela “Atividade”.

idAtividade	designacao
1	Construção civil
2	Pintura
3	Eletrotécnica
4	Carpintaria
5	Serviços

Tabela 6 - Povoamento da tabela “Questao”.

idQuestao	Descricao	Ativa
1	Esta disposto a trabalhar connosco?	true
2	Quais os problemas da casa actual?	true
3	Tem terreno?	true
4	Durante a obra, terá onde ficar?	true
5	Em que dias estaria disponível para trabalhar na obra?	true

Tabela 7 - Povoamento da tabela “TipoDonativo”.

idTipoDon	designação	unidade
1	Monetario	€
2	Cimento	sacos
3	Tijolo	Kg
4	Areia	Kg
5	Telha	m2

Tabela 8 - Povoamento da tabela “Individuo” (foram omitidos alguns atributos).

idIndiv	idFunc	nome	dataNascimento	isDoador	isVoluntario
1	3	Suki Swanson	1977-10-16	1	0
2	3	Germane Gilbert	1969-10-09	0	1
3	3	Basil P. Barry	1963-10-27	1	1
4	3	Heather Pierce	1968-10-16	0	1
5	3	Edu Severina	1988-04-20	0	1

Tabela 9 - Povoamento da tabela “Familia” (foram omitidos alguns atributos).

idFam	nome	apeldo	dataCriaFam	idFunc
1	Tamara Brenna	Brenna	2009-06-06	2
2	Gregory Karen	Karen	2008-1-05	2
3	Nichole roanna	Roanna	2007-12-11	2
4	Yeo Sigourney	Sigourney	2007-03-17	2

idFam	nome	apeldo	dataCriaFam	idFunc
5	Adelaide Francisco	Francisco	2001-08-5	2
6	Jônatas Tristão	Tristão	2011-08-5	2

Tabela 10 - Povoamento da tabela “ElementoFamilia” (foram omitidos alguns atributos).

idElemento	idFam	nome	parentesco
1	1	Lana Cosby	sobrnho
2	1	Quincy Cannon	pai
3	2	Josephine Estrada	pai
4	2	Sybill Velasquez	primo
5	3	Patience Moody	primo
6	4	Amaya Potter	filho

Tabela 11 - Povoamento da tabela “Candidatura” (foram omitidos alguns atributos).

idCand	idFam	dataCand	rendimento	estado
1	1	2014-05-11	2000	APROVADO
2	1	2012-08-10	720	NAOAPROVADO
3	2	2013-12-24	1200	APROVADO
4	3	2013-12-28	776	APROVADO
5	4	2014-09-28	744	APROVADO
6	5	2015-11-19	888	PENDENTE
7	6	2014-12-29	422	PENDENTE

Tabela 12 - Povoamento da tabela “CandidaturaQuestao” (foram omitidos alguns atributos).

idCand	idQuestao	resposta
1	1	Não
1	2	Telhado Destruido
1	3	Sim
1	5	Nunca
2	1	Sim
2	2	Idade avançada
2	3	Não
2	4	Sim

Tabela 13 - Povoamento da tabela “Projeto” (foram omitidos alguns atributos)

idProj	orcamento	prazo	idCand	designacao	dataCriaProj
1	123000	2016-06-29	1	Reparação do telhado	2014-12-21
2	136000	2015-01-30	3	Reparações eletricas	2014-12-21

idProj	orcamento	prazo	idCand	designacao	dataCriaProj
3	145000	2016-01-05	4	Reparações da Canalização	2014-12-21
4	154000	2016-03-26	5	Construçao de casa nova	2014-12-21

Tabela 14 - Povoamento da tabela “TarefaProjeto” (foram omitidos alguns atributos)

idProj	idTar	dataPLInicio	dataPLFim
1	1	2014-10-23	2014-10-15
1	2	2014-10-12	2014-10-15
2	3	2014-11-10	2014-10-15
2	4	2014-10-19	2014-10-15
2	5	2014-10-20	2014-10-15
3	1	2014-03-09	2014-02-11

Tabela 15 - Povoamento da tabela “Evento”

idEv	designacao	data	idFunc	Descricao
1	Evento espontâneo	DATE(NOW())	1	Evento usado para donativos gerais
2	Angariação de fundos do Sr. Luis	2014-02-23	1	Evento usado para donativos de materiais
3	Dia de Natal	2013-12-20	1	Evento usado para donativos monetarios

Tabela 16 - Povoamento da tabela “IndividuoTarefaProjeto” (foram omitidos alguns atributos)

idIndiv	dataTrabalho	duracao	idProj	idTar
2	2014-03-12	12	1	1
2	2014-03-23	24	3	2
3	2014-02-10	43	2	3
5	2014-02-13	55	2	4
5	2014-03-04	4	2	5
3	2014-03-18	46	3	1

Tabela 17 - Povoamento da tabela “IndividuoAtividade”

idIndiv	idAtividade
1	4
2	1
3	2
4	2
5	4
1	2
5	2

Tabela 18 - Povoamento da tabela “Equipa”

idEq	nacionalidadeEq	designacao	dataCriaEq	idFunc
1	Portuguesa	Empresa UnknownX	NOW()	3
2	Italiana	Ferias de voluntariado	NOW()	3
3	Inglesa	Associação Vicamonesis	NOW()	3

Tabela 19 - Povoamento da tabela “EquipaIndividuo”

idIndiv	idEq
2	1
4	1
3	2
5	3

Tabela 20 - Povoamento da tabela “EquipaTarefaProjeto”

idEq	dataTrabalho	duracao	idProj	idTar
1	2014-03-15	5	1	2
2	2014-03-02	2	2	4
3	2014-03-22	12	2	5

Tabela 21 - Povoamento da tabela “Donativo” (foram omitidos alguns atributos)

idDon	quantInicial	quantAtual	idTipoDon	idFunc	idEv	idIndiv
1	10	6	1	3	1	1
2	10	10	1	3	1	1
3	10	9	3	3	1	3
5	10	6	1	3	1	1

Tabela 22 - Povoamento da tabela “DonativoProjeto”

idDon	idProj	dataDonProj	quantDonProj	idFunc
1	1	2014-10-29	4	3
3	2	2014-11-12	1	3
5	2	2014-10-29	4	3

Tabela 23 - Povoamento da tabela “Relatorio” (foram omitidos alguns atributos)

obsRel	dataRel	idFam	idFunc
Familia a habituar-se normalmente	2014-12-21	1	1
Falta de limpeza externa na casa	2014-12-21	2	1

Tabela 24 - Povoamento da tabela “Prestacao”

idFunc	valor	data	idFam
2	100	2012-10-21	1
2	120	2014-12-21	1
2	130	2014-12-21	2
2	120	2014-12-21	3
2	100	2014-12-21	4

6.7. Estimativa do Espaço de Disco Necessário e crescimento da BD

Após o povoamento da BD, descrito no item anterior, foi determinado o espaço ocupado e o tamanho médio de cada linha, ou entrada nas tabelas. O espaço de disco ocupado pelas tabelas foi determinado considerando tanto o tamanho dos dados com dos índices. Para o cálculo de todas as tabelas foi utilizada a script abaixo exposta. Os resultados obtidos encontram-se descritos nas quatro colunas iniciais da Tabela 25.

```
SELECT table_name 'Tabela',
         round(((data_length + index_length) / 1024), 0) 'Tamanho (kB)',
         table_rows 'No. linhas',
       round((data_length + index_length)/table_rows,0) 'Tamanho por linha (bytes)'
FROM information_schema.TABLES
WHERE table_schema = "habitat";
```

Na ausência de dados mais precisos, realizou-se, utilizando o bom senso e o reduzido conhecimento amealhado, uma estimativa do número de entradas iniciais e taxas de crescimento anual para cada tabela. Com base nestas estimativas, extrapolaram-se os valores para um horizonte de dez anos, tendo-se obtido um tamanho de cerca de 44 MB. Aplicando um fator de segurança de 2, de forma a ter em conta as incertezas inerentes às estimativas realizadas, define-se um espaço necessário de cerca de 90 MB.

Como esperado, estima-se que a BD venha a ter uma dimensão reduzida e como tal o espaço de disco necessário não constitui um condicionamento importante. O armazenamento de ficheiros digitalizados de candidaturas ou fichas de registo poderá ter um impacto significativo no espaço necessário. Este aspeto não foi tido em conta na estimativa realizada.

Tabela 25 – Estimativa do crescimento da BD.

Tabela	Tamanho (kB) (*)	No. linhas (*)	Tamanho por linha (bytes) (*)	No. linhas inicial (**)	Tamanho inicial (kB) (**)	Taxa de Crescim. Anual (**)	Tamanho estimado a 10 anos (kB) (**)
actividade	16	5	3277	10	32	10%	83
candidatura	32	7	4681	15	69	20%	425
candidaturaquestao	48	17	2891	300	847	20%	5244
donativo	80	4	20480	30	600	30%	8272
donativoprojeto	64	3	21845	60	1280	30%	17646
elementofamilia	32	15	2185	40	85	20%	528
equipa	32	3	10923	5	53	20%	330
equipaindividuo	48	4	12288	10	120	20%	743
equipatarefaprojeto	48	3	16384	30	480	20%	2972
evento	32	3	10923	5	53	20%	330
familia	48	6	8192	10	80	20%	495
funcionario	32	4	8192	4	32	10%	83
individuo	48	5	9830	20	192	20%	1189
individuoactividade	48	7	7022	20	137	20%	849
individuotarefaprojeto	48	10	4915	30	144	20%	892
prestacao	48	5	9830	5	48	20%	297
projeto	48	4	12288	5	60	20%	372
questao	16	5	3277	20	64	10%	166
relatorio	48	2	24576	3	72	20%	446
tarefa	16	7	2341	20	46	10%	119
tarefaprojeto	48	10	4915	100	480	20%	2972
tipodonativo	16	5	3277	10	32	10%	83
				TOT=	**5006**	**TOT=**	**44535**

Notas:
(*) Com base no povoamento realizado
(**) Valores estimados

6.8. Validação com Transações do Utilizador

Seguidamente apresentação as scripts para realização das *queries* definidas no capítulo 2 - Levantamento e Análise de Requisitos.

Junto a cada *querie* apresenta-se também o output obtido, após o povoamento já descrito.

Transação 1: Listar os projetos para os quais um doador contribui.

```
SELECT DISTINCT PR.idProj, PR.orcamento, PR.designacao, PR.descricao
        FROM Projeto AS PR
                INNER JOIN DonativoProjeto AS DP
                        ON DP.idProj = PR.idProj
```

```
        INNER JOIN Donativo AS DN
                ON DN.idDon = DP.idDon
    WHERE DN.idIndiv = 1; -- Individuo 1: Suki Swanson
```

Tabela 26 – Output querie 1.

idProj	orcamento	designacao	descricao
1	123000	Reparação do telhado	Reparar o telhado para evitar a chuva
2	136000	Reparacões elétricas	Reconstruir instalação elétrica

Transação 2: Listar os projetos para os quais um voluntário trabalhou individualmente.

```
SELECT DISTINCT PR.idProj, PR.orcamento, PR.designacao, PR.descricao
    FROM Projeto AS PR
        INNER JOIN IndividuoTarefaProjeto AS ITP
            ON ITP.idProj = PR.idProj
    WHERE ITP.idIndiv = 2; -- Individuo 2: Germane Gilbert
```

Tabela 27 – Output querie 2.

idProj	orcamento	designacao	Descrição
1	123000	Reparação do telhado	Reparar o telhado para evitar a chuva
3	145000	Reparacões da Canalização	Reconstruir instalação elétrica
4	154000	Construção de casa nova	Construir casa de raiz para família desalojada

Transação 3: Listar os donativos alocados a um determinado projeto.

```
SELECT DISTINCT DN.idDon, DN.descricao, DN.quantInicial, DN.dataDon
    FROM Donativo AS DN
        INNER JOIN DonativoProjeto as DP
            ON DP.idDon = DN.idDon
    WHERE DP.idProj = 2; -- Projeto 2: Reparações eletricas
```

Tabela 28 – Output querie 3.

idDon	descricao	quantInicial	dataDon
3	Doação de material em excesso	10	2014-11-26

5	Doação Monetária	10	2014-09-23

Transação 4: Identificar a família a que corresponde um determinado projeto.

```
SELECT FA.idFam, FA.apelido, FA.dataCriaFam, FA.nome, FA.telefone
	FROM Familia AS FA
		INNER JOIN Candidatura AS CA
			ON CA.idFam = FA.idFam
		INNER JOIN Projeto AS PR
			ON PR.idCand = CA.idCand
	WHERE PR.idProj = 1; -- Projeto 1: Reparação do telhado
```

Tabela 29 – Output querie 4.

idFam	apelido	dataCriaFam	nome	telefone
1	Brenna	2009-06-06	Tamara Brenna	8690220

Transação 5: Listar os relatórios realizados para uma dada família.

```
SELECT * FROM relatorio
	WHERE idFam = 1; -- Familia: Brenna
```

Tabela 30 -– Output querie 5.

idRel	ObsRel	dataRel	ficheiroRel	idFam	idFunc
1	Familia a habituar-se normalmente	2014-12-21	null	1	1

Transação 6: Listar o número de doadores por atividade.

```
SELECT AI.idAtividade, AI.designacao, count(AI.idAtividade) AS Total
	FROM Actividade AS AI
		INNER JOIN IndividuoActividade AS IAC
			ON IAC.idAtividade = AI.idAtividade
		INNER JOIN Individuo AS ID
			ON ID.idIndiv = IAC.idIndiv
	WHERE isDoador = true
	GROUP BY AI.idAtividade, AI.designacao;
```

Tabela 31 – Output querie 6.

idAtividade	Designacao	Total
2	Pintura	2
4	Carpintaria	1

7. Conclusões e Trabalho Futuro

No trabalho prático desenvolvido foram devidamente documentadas as etapas do modelo conceptual, lógico e físico, seguindo a metodologia relacional proposta por Connolly & Begg (2002). Com base na documentação criada é possível obter um conhecimento detalhado do trabalhado realizado e, porventura, continuar o trabalhado implementando eventuais melhorias e ajustes.

O trabalho foi desenvolvido procurando sempre respeitar os requisitos do utilizador final. No entanto, os requisitos não chegaram a ser formulados ou confirmados por escrito pelo requerente, tendo sido definidos pelos autores com base nos vários elementos recolhidos.

Entre a etapa inicial das reuniões com o representante da Habitat até a etapa final da implementação da base de dados, foram realizadas várias iterações procurando uma melhoria contínua e dos modelos. Pese embora se considere que seria conveniente uma ulterior validação com a Habitat, admite-se que os objetivos foram na sua maioria alcançados. Os resultados obtidos através das várias validações realizadas, incluindo o povoamento e a realização de *queries* realiza, forneceram boas indicações sobre a qualidade e correção dos modelos desenvolvidos.

Por fim, apesar do trabalho da conclusão do trabalho, considera-se que poderão ainda ser feitas melhorias, que se listam seguidamente.

Trabalho futuro:

- Nova iteração com a Habitat para melhor definição mais *queries* e confirmação dos requisitos.
- Povoamento com dados reais da Habitat para uma melhor compreensão dos resultados obtidos.
- Melhoria na atomicidade de alguns atributos, como o nome ou morada.
- Melhor implementação das transações, implementando procedimentos (*stored procedures*) e ultrapassando as dificuldades de argumentos multi-valor (*array*).
- Melhoria das políticas de backup, de acordo com requisitos da Habitat.

Bibliografia

CHEN, P. P.-S. (1976), The Entity-Relationship Model - Toward a unified View of Data, in , ACM, New York, NY, USA , pp. 9-36 .

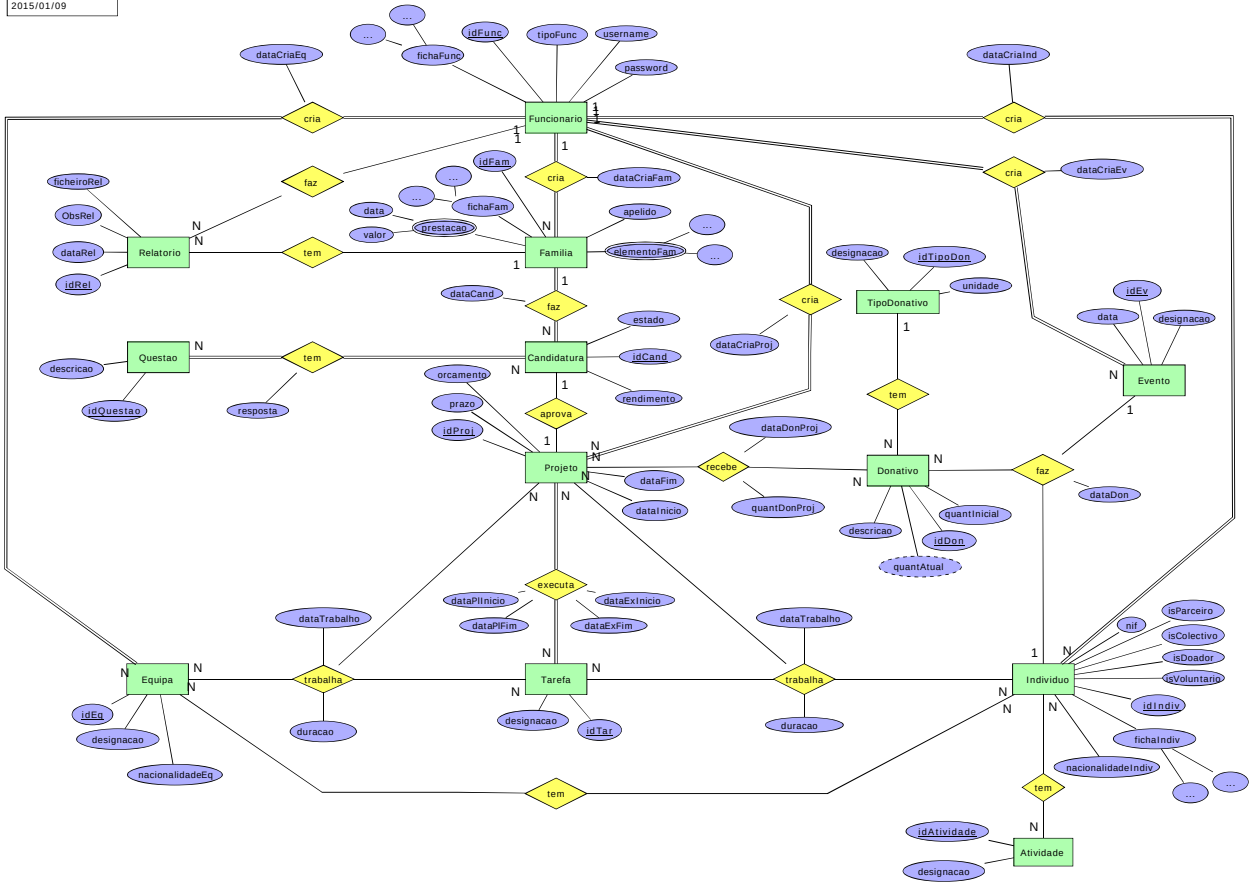
CONNOLLY, T. M., & BEGG, C. E. (2002). Database systems: a practical approach to design, implementation, and management. Harlow, England, Addison-Wesley.

Lista de Siglas e Acrónimos

BD	Base de Dados
SGBD	Sistemas de Gestão de Bases de Dados

I. Anexo – Diagrama Conceptual

Projeto Habitat
Grupo 24
Modelo conceptual
habitat-concep-v8
2015/01/09



Notas:
- Por uma questão de facilidade de leitura, não se representam todos os atributos envolvidos.
- Os atributos "fichaFam", "fichaIndiv" e "fichaFunc" são virtuais e representam, no presente modelo, a totalidade da informação de descrição e contacto das respectivas entidades.

II. Anexo – Diagrama Lógico

Comissão de Fundos

Comissão de Obras

Administrador

Comissão de Famílias

Projeto Habitat
Grupo 24
Modelo lógico
habitat-logico-v1.16
2015/01/09

Donativo
- idDon INT
- descricao VARCHAR(100)
- quantInicial INT
- quantAtual INT
- idTipoDon INT
- dataDon DATE
- idFunc INT
- idEv INT
- idIndiv INT

Indexes

Individuo
- idIndiv INT
- idFunc INT
- nome VARCHAR(75)
- dataNascimento DATE
- profissao VARCHAR(45)
- morada VARCHAR(75)
- codigoPostal VARCHAR(10)

20 more...

Indexes

DonativoProjeto
- idDon INT
- idProj INT
- dataDonProj DATE
- quantDonProj INT
- idFunc INT

Indexes
Triggers

EquipaIndividuo
- idIndiv INT
- idEq INT

Indexes

IndividuoTarefaProjeto
- idIndiv INT
- dataTrabalho DATE
- duracao INT
- idProj INT
- idTar INT

Indexes

EquipaTarefaProjeto
- idEq INT
- dataTrabalho DATE
- duracao INT
- idProj INT
- idTar INT

Indexes

Evento
- idEv INT
- designacao VARCHAR(100)
- data DATE
- idFunc INT
- descricao VARCHAR(200)

Indexes

IndividuoActividade
- idIndiv INT
- idAtividade INT

Indexes

Equipa
- idEq INT
- nacionalidadeEq VARCHAR(45)
- designacao VARCHAR(100)
- dataCriaEq DATETIME
- idFunc INT

Indexes

TarefaProjeto
- idProj INT
- idTar INT
- dataPlInicio DATETIME
- dataPlFim DATETIME
- dataExInicio DATETIME
- dataExFim DATETIME

Indexes

Projeto
- idProj INT
- orcamento INT
- prazo DATE
- dataInicio DATE
- dataFim DATE
- idFunc INT
- idCand INT
- designacao VARCHAR(75)
- descricao VARCHAR(200)
- dataCriaProj DATE

Indexes

ViewTarefaPro...

Actividade
- idAtividade INT
- designacao VARCHA...

Indexes

Tarefa
- idTar INT
- designacao VARCHAR(45)

Indexes

Questao
- idQuestao INT
- descricao VARCHAR(200)
- ativa BOOLEAN

Indexes

TipoDonativo
- idTipoDon INT
- designacao VARCHAR(50)
- unidade VARCHAR(20)

Indexes

Funcionario
- idFunc INT
- tipoFunc ENUM(...)
- nome VARCHAR(75)
- nif INT
- morada VARCHAR(65)
- telefone VARCHAR(20)
- telemovel VARCHAR(20)
- codigoPostal VARCHAR(10)
- dataNascimento DATE
- localidade VARCHAR(45)
- password VARCHAR(45)
- username VARCHAR(45)

Indexes

CandidaturaQues...
- idCand INT
- idQuestao INT
- resposta VARCHAR(100)

Indexes

ViewCandidaturaQues...

Candidatura
- idCand INT
- idFam INT
- dataCand DATE
- rendimento DECIMAL(1...
- estado ENUM(...)

Indexes

Prestacao
- idPrestacao INT
- idFunc INT
- valor DECIMAL(10,2)
- data DATE
- idFam INT

Indexes

Familia
- idFam INT
- nome VARCHAR(45)
- morada VARCHAR(45)
- telefone INT
- idFunc INT
- nif INT
- dataNascimento DATE
- apelido VARCHAR(45)
- dataCriaFam DATE

Indexes

Relatorio
- idRel INT
- obsRel VARCHAR(45)
- dataRel DATE
- ficheiroRel BLOB
- idFam INT
- idFunc INT

Indexes

ElementoFamilia
- idElemento INT
- idFam INT
- nome VARCHAR(75)
- parentesco VARCHAR(20)
- dataNascimento DATE
- estadoCivil VARCHAR(20)
- ocupacao VARCHAR(55)
- escolaridade VARCHAR(20)

Indexes

III. Anexo – Dicionário de Dados

Legenda:
PK - chave primária (primary key)
FK - chave estrangeira (foreign key)
NN - campo não nulo (not null)
DA - atributo derivado (derived attribute)
DV - valor por omissão (default value)
AT - valor auto-numerado (auto-numbered)

Layers:
Comissão de fundos
Comissão de obras
Comissão de famílias
Administrator

Tabela	Alias	Descrição	Tipo de Dados	PK	FK	NN	DA	DV	AT
Donativo	idDon	Valor identificativo de cada donativo	INT	x		x			x
	descricao	Descrição do donativo	VARCHAR(100)						
	quantInicial	Quantidade inicial doada	INT			x			
	quantAtual	Quantidade atualmente disponível do donativo inicial	UNSIGNEDINT			x	x		
	idTipoDon	Valor identificativo do tipo de donativo	INT		x	x			
	dataDon	Data em que foi feito o donativo	DATE						
	idFunc	Valor identificativo do funcionário que recebeu o donativo	INT		x	x			
	idEv	Valor identificativo do evento em que a doação foi feita	INT		x	x			
	idIndiv	Valor identificativo do doador	INT		x	x			
Individuo	idIndiv	Valor identificativo do indivíduo	INT	x		x			x
	idFunc	Valor identificativo do funcionário que registou o indivíduo	INT		x	x			
	nome	Nome do indivíduo	VARCHAR(75)			x			
	dataNascimento	Data de nascimento do indivíduo	DATE						
	profissao	Profissão do indivíduo	VARCHAR(45)						
	morada	Morada do indivíduo	VARCHAR(75)						
	codigoPostal	Código postal do indivíduo	VARCHAR(10)						
	localidade	Localidade onde reside o indivíduo	VARCHAR(60)						
	telefone	Número de telefone do indivíduo	VARCHAR(20)						
	telemovel	Número de telemóvel do indivíduo	VARCHAR(20)						
	email	Email do indivíduo	VARCHAR(45)						
	habilitacoes	Habilitações literárias do indivíduo	VARCHAR(45)						
	conhecimentosLing	Conhecimentos linguísticos do indivíduo	VARCHAR(45)						
	formacaoComp	Formação complementar do indivíduo	VARCHAR(45)						
	experienciaVolunt	Experiência de voluntariado do indivíduo	VARCHAR(45)						
	conhecimentosContr	Conhecimentos de construção do indivíduo	VARCHAR(45)						
	trabalharJuntoVolunt	Se o indivíduo deseja trabalhar junto de outros voluntários	BOOLEAN						
	disponibilidade	Disponiblidade de tempo do indivíduo	VARCHAR(75)						
	comoConheceu	Como o indivíduo tomou conhecimento da habitat	VARCHAR(45)						
	receberInfo	Se o indivíduo deseja receber informações sobre a habitat	BOOLEAN						
	isParceiro	Se o indivíduo é parceiro da habitat	BOOLEAN		x	x		FALSE	
	nif	NIF do indivíduo	INT						
	isColetivo	Se o "indivíduo" se trata de um grupo ou empresa	BOOLEAN		x	x		FALSE	
	isDoador	Se o indivíduo é doador	BOOLEAN		x	x		FALSE	
	isVoluntario	Se o indivíduo é voluntário	BOOLEAN		x	x		FALSE	
	nacionalidadeIndiv	Nacionalidade do indivíduo	VARCHAR(45)						
	dataCriaIndiv	Data de registo do indivíduo	DATE						
EquipaIndividuo	idIndiv	Valor identificativo do indivíduo que faz parte da equipa	INT	x	x	x			
	idEq	Valor identificativo da equipa	INT	x	x	x			
Equipa	idEq	Valor identificativo da equipa	INT	x		x			x
	nacionalidadeEq	Nacionalidade da equipa	VARCHAR(45)						
	designacao	Designação da equipa	VARCHAR(100)			x			
	dataCriaEq	Data de criação da equipa	DATE			x			
	idFunc	Valor identificativo do funcionário que registou a equipa	INT		x	x			
Evento	idEv	Valor identificativo do evento	INT	x		x			x
	designacao	Designação do evento	VARCHAR(100)			x			
	data	Data do evento	DATE						
	idFunc	Valor identificativo do funcionário que registou o evento	INT		x	x			
	descricao	Descrição do evento	VARCHAR(200)						
IndividuoActividade	idIndiv	Valor identificativo do indivíduo que realiza a atividade	INT	x	x	x			
	idAtividade	Valor identificativo da atividade	INT	x	x	x			
	idDon	Valor identificativo do donativo	INT	x	x	x			

Legenda:
PK - chave primária (primary key)
FK - chave estrangeira (foreign key)
NN - campo não nulo (not null)
DA - atributo derivado (derived attribute)
DV - valor por omissão (default value)
AT - valor auto-numerado (auto-numbered)

Layers:
Comissão de fundos
Comissão de obras
Comissão de famílias
Administrator

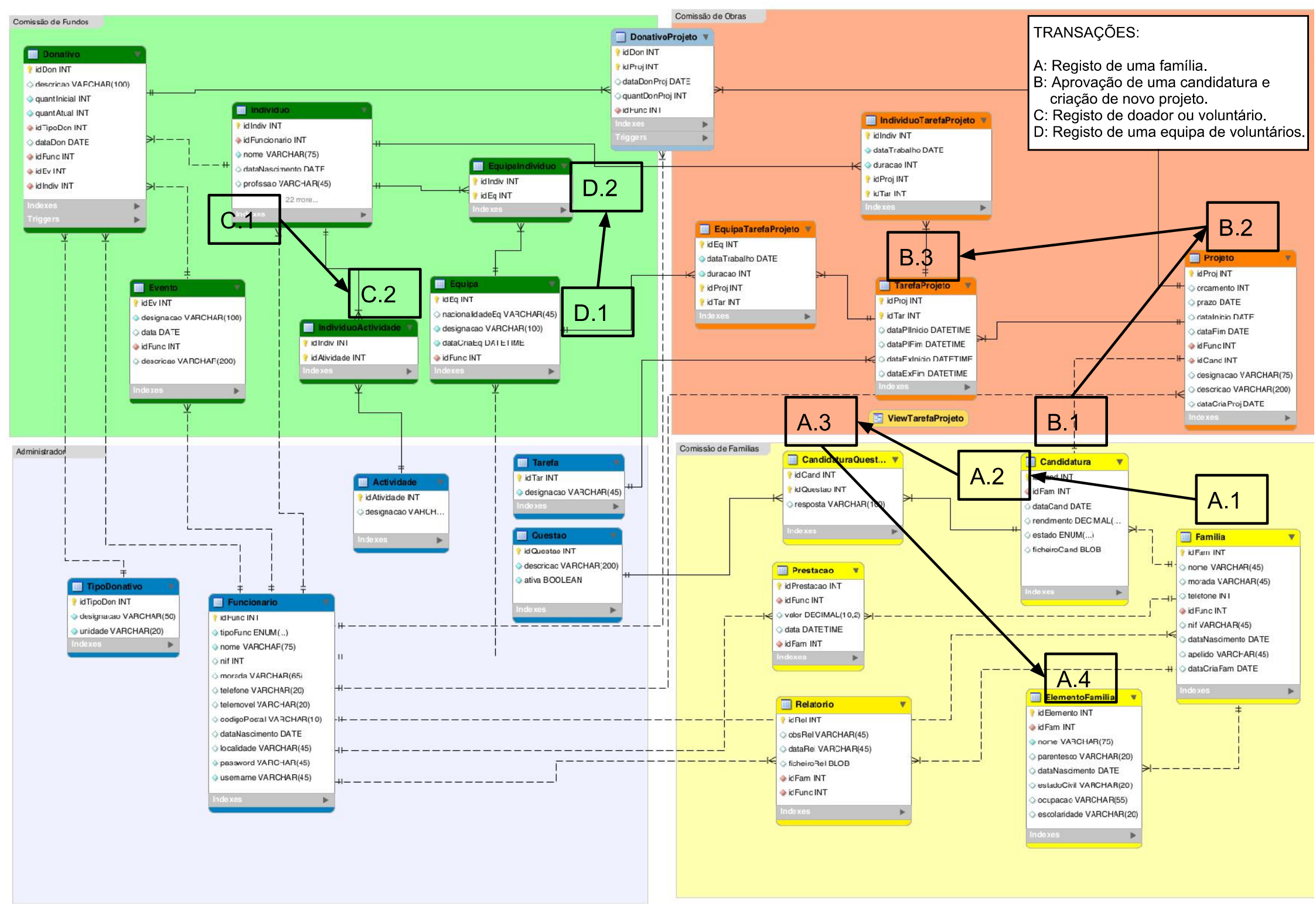
Tabela	Alias	Descrição	Tipo de Dados	PK	FK	NN	DA	DV	AT
DonativoProjeto	idProj	Valor identificativo do projeto para o qual o donativo é feito	INT	x	x	x			
	dataDonProj	Data da doação	DATE						
	quantDonProj	Quantidade doada	INT						
	idFunc	Valor identificativo do funcionário que registou a doação	INT			x			
IndividuoTarefaProjeto	idIndiv	Valor identificativo do indivíduo	INT	x	x	x			
	dataTrabalho	Data em que foi registada a tarefa	DATE			x			
	duracao	Duração da tarefa	INT			x			
	idProj	Valor identificativo do projeto	INT	x	x	x			
	idTar	Valor identificativo da tarefa	INT	x	x	x			
EquipaTarefaProjeto	idEq	Valor identificativo da tarefa	INT	x	x	x			
	dataTrabalho	Data em que foi registada a tarefa	DATE			x			
	duracao	Duração da tarefa	INT			x			
	idProj	Valor identificativo do projeto	INT	x	x	x			
	idTar	Valor identificativo da tarefa	INT	x	x	x			
TarefaProjeto	idProj	Valor identificativo do projeto	INT	x	x	x			
	idTar	Valor identificativo da tarefa	INT	x	x	x			
	dataPlInicio	Data planeada para início da tarefa	DATE						
	dataPlFim	Data planeada para fim da tarefa	DATE						
	dataExInicio	Data efetiva de início da tarefa	DATE						
	dataExFim	Data efetiva de fim da tarefa	DATE						
Projeto	idProj	Valor identificativo do projeto	INT	x		x			x
	orcamento	Orçamento do projeto	INT						
	prazo	Prazo estimado do projeto (data de conclusão estimada)	DATE						
	dataInicio	Data de início do projeto	DATE						
	dataFim	Data de fim do projeto	DATE						
	idFunc	Valor identificativo do funcionário responsável pelo projeto	INT		x	x			
	idCand	Valor identificativo da cadidatura respetiva ao projeto	INT		x	x			
	designacao	Designação do projeto	VARCHAR(75)						
	descricao	Descrição do projeto	VARCHAR(200)						
	dataCriaProj	Data de criação do projeto	DATE						
Familia	idFam	Valor identificativo da família	INT	x		x			x
	nome	Nome do representante da família	VARCHAR(45)						
	morada	Morada do representante da família	VARCHAR(45)						
	telefone	Número de telefone do representante da família	INT						
	idFunc	Valor identificativo do funcionário responsável pela candidatura	INT			x			
	nif	NIF do representante da família	INT						
	dataNascimento	Data de nascimento do representante da família	DATE						
	apelido	Apelido do representante da família	VARCHAR(45)						
	dataCriaFam	Data de registo da famíia	DATE						
Candidatura	idCand	Valor identificativo da candidatura	INT	x		x			x
	idFam	Valo identificativo da família correspondente à candidatura	INT		x	x			
	dataCand	Data em que foi feita a candidatura	DATE						
	rendimento	Rendimento anual bruto da família	DECIMAL(10,2)						
	estado	Estado da candidatura: 'APROVADO': A candidatura foi aprovada e o projeto aceite pela familia; 'NAOAPROVADO': a candidatura não foi aprovada; 'NAOACEITE': a candidatura foi aprovada mas o prjeto não foi aceite pela família; 'PENDENTE': a candidatura aguarda aprovamento.	ENUM(...)			x		'PENDENTE'	
CandidaturaQuestao	idCand	Valor identificativo da candidatura	INT	x	x	x			
	idQuestao	Valor identificativo da questão	INT	x	x	x			
	resposta	Resposta à questão	VARCHAR(100)						

Legenda:
PK - chave primária (primary key)
FK - chave estrangeira (foreign key)
NN - campo não nulo (not null)
DA - atributo derivado (derived attribute)
DV - valor por omissão (default value)
AT - valor auto-numerado (auto-numbered)

Layers:
Comissão de fundos
Comissão de obras
Comissão de famílias
Administrator

Tabela	Alias	Descrição	Tipo de Dados	PK	FK	NN	DA	DV	AT
ElementoFamilia	idElemento	Valor identificativo do elemento de uma familia	INT	x		x			x
	idFam	Valor identificativo da família à qual o elemento pertence	INT		x	x			
	nome	Nome do elemento	VARCHAR(75)			x			
	parentesco	Parentesco do elemento em relação ao representante	VARCHAR(20)						
	dataNascimento	Data de nascimento do elemento	DATE						
	estadoCivil	Estado civil do elemento	VARCHAR(20)						
	ocupacao	Ocupação do elemento	VARCHAR(55)						
	escolaridade	Habilitações literárias do elemento	VARCHAR(20)						
Relatorio	idRel	Valor identificativo do relatório	INT	x		x			x
	obsRel	Observações sobre o relatório	VARCHAR(45)						
	dataRel	Data do relatório	DATE						
	ficheiroRel	Ficheiro digitalizado do relatório	BLOB						
	idFam	Valor identificativo da família à qual o relatório de refere	INT		x	x			
	idFunc	Valor identificativo do funcionário responsável pelo relatório	INT		x	x			
Prestacao	idPrestacao	Valor identificativo da prestação	INT	x		x			x
	idFunc	Valor identificativo do funcionário responsável	INT		x	x			
	valor	Valor da prestação a pagar	DECIMAL(10,2)						
	data	Data em que foi definida a prestação	DATE						
	idFam	Valor identificativo da família a pagar a prestação	INT		x	x			
Funcionario	idFunc	Valor identificativo do funcionário	INT	x		x			x
	tipoFunc	Tipo de funcionário: 'ADMIN': é um administrador; 'FAM': trabalha na comissão de famílias; 'FUNDOS':trabalha na comissão de fundos; 'OBRAS': trabalha na comissão de obras.	ENUM(...)			x			
	nome	Nome do funcionário	VARCHAR(75)			x			
	nif	NIF do funcionário	INT						
	morada	Morada do funcionário	VARCHAR(65)						
	telefone	Número de telefone do funcionário	VARCHAR(20)						
	telemovel	Número de telemóvel do funcionário	VARCHAR(20)						
	codigoPostal	Codigo postal do funcionário	VARCHAR(10)						
	dataNascimento	Data de nascimento do funcionário	DATE						
	localidade	Localidade onde reside o funcionário	VARCHAR(45)						
	password	Password de acesso do funcionário	VARCHAR(45)			x			
	username	Username de acesso do funcionário	VARCHAR(45)			x			
TipoDonativo	idTipoDon	Valor identificativo do tipo de donativo	INT	x		x			x
	designacao	Designação do tipo de donativo	VARCHAR(50)			x			
	unidade	Unidades relativa ao tipo de donativo	VARCHAR(20)			x			
Actividade	idAtividade	Valor identificativo da atividade	INT	x		x			x
	designacao	Designação da atividade	VARCHAR(55)						
Tarefa	idTar	Valor identificativo da tarefa	INT	x		x			x
	designacao	Designação da tarefa	VARCHAR(45)			x			
Questao	idQuestao	Valor identificativo da questão	INT	x		x			x
	descricao	Texto da questão a formular	VARCHAR(200)						
	ativa	Se a questão se encontra ativa	BOOLEAN			x			

IV. Anexo – Mapa de Transações


Comissão de Fundos
Comissão de Obras
Administrador
Comissão de Famílias
TRANSAÇÕES:
A: Registo de uma família.
B: Aprovação de uma candidatura e criação de novo projeto.
C: Registo de doador ou voluntário.
D: Registo de uma equipa de voluntários.
Donativo
idDon INT
descricao VARCHAR(100)
quantInicial INT
quantAtual INT
idTipoDon INT
dataDon DATE
idFunc INT
idEv INT
idIndiv INT
DonativoProjeto
idDon INT
idProj INT
dataDonProj DATE
quantDonProj INT
idFunc INT
Individuo
idIndiv INT
idFuncionario INT
nome VARCHAR(75)
dataNascimento DATE
profissao VARCHAR(45)
22 more...
EquipaIndividuo
idIndiv INT
idEq INT
IndividuoTarefaProjeto
idIndiv INT
dataTrabalho DATE
duracao INT
idProj INT
idTar INT
EquipaTarefaProjeto
idEq INT
dataTrabalho DATE
duracao INT
idProj INT
idTar INT
Evento
idEv INT
designacao VARCHAR(100)
data DATE
idFunc INT
descricao VARCHAR(200)
IndividuoActividade
idIndiv INT
idAtividade INT
Equipa
idEq INT
nacionalidadeEq VARCHAR(45)
designacao VARCHAR(100)
dataCriaEq DATETIME
idFunc INT
TarefaProjeto
idProj INT
idTar INT
dataPlInicio DATETIME
dataPlFim DATETIME
dataExInicio DATETIME
dataExFim DATETIME
Projeto
idProj INT
orcamento INT
prazo DATE
dataInicio DATE
dataFim DATE
idFunc INT
idCand INT
designacao VARCHAR(75)
descricao VARCHAR(200)
dataCriaProj DATE
ViewTarefaProjeto
Actividade
idAtividade INT
designacao VARCH...
Tarefa
idTar INT
designacao VARCHAR(45)
Questao
idQuestao INT
descricao VARCHAR(200)
ativa BOOLEAN
CandidaturaQuest...
idCand INT
idQuestao INT
resposta VARCHAR(100)
Candidatura
idCand INT
idFam INT
dataCand DATE
rendimento DECIMAL(...
estado ENUM(...)
ficheiroCand BLOB
Familia
idFam INT
nome VARCHAR(45)
morada VARCHAR(45)
telefone INT
idFunc INT
nif VARCHAR(45)
dataNascimento DATE
apelido VARCHAR(45)
dataCriaFam DATE
TipoDonativo
idTipoDon INT
designacao VARCHAR(50)
unidade VARCHAR(20)
Funcionario
idFunc INT
tipoFunc ENUM(...)
nome VARCHAR(75)
nif INT
morada VARCHAR(65)
telefone VARCHAR(20)
telemovel VARCHAR(20)
codigoPostal VARCHAR(10)
dataNascimento DATE
localidade VARCHAR(45)
password VARCHAR(45)
username VARCHAR(45)
Prestacao
idPrestacao INT
idFunc INT
valor DECIMAL(10,2)
data DATETIME
idFam INT
Relatorio
idRel INT
obsRel VARCHAR(45)
dataRel VARCHAR(45)
ficheiroRel BLOB
idFam INT
idFunc INT
ElementoFamilia
idElemento INT
idFam INT
nome VARCHAR(75)
parentesco VARCHAR(20)
dataNascimento DATE
estadoCivil VARCHAR(20)
ocupacao VARCHAR(55)
escolaridade VARCHAR(20)
Indexes
Triggers
C.1
C.2
D.1
D.2
B.1
B.2
B.3
A.1
A.2
A.3
A.4

V. Anexo – Script de Criação da BD

```
-- MySQL Script generated by MySQL Workbench
-- 01/09/15 18:51:06
-- Model: habitat-logico    Version: 1.16
-- 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';

-- -----------------------------------------------------
-- Schema habitat
-- -----------------------------------------------------
DROP SCHEMA IF EXISTS `habitat` ;

-- -----------------------------------------------------
-- Schema habitat
-- -----------------------------------------------------
CREATE SCHEMA IF NOT EXISTS `habitat` DEFAULT CHARACTER SET utf8 COLLATE utf8_general_ci ;
USE `habitat` ;

-- -----------------------------------------------------
-- Table `habitat`.`Funcionario`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `habitat`.`Funcionario` ;

CREATE TABLE IF NOT EXISTS `habitat`.`Funcionario` (
  `idFunc` INT NOT NULL AUTO_INCREMENT,
  `tipoFunc` ENUM('ADMIN','FAM','FUNDOS','OBRAS') NOT NULL,
  `nome` VARCHAR(75) NOT NULL,
  `nif` INT NULL,
  `morada` VARCHAR(65) NULL,
  `telefone` VARCHAR(20) NULL,
  `telemovel` VARCHAR(20) NULL,
  `codigoPostal` VARCHAR(10) NULL,
  `dataNascimento` DATE NULL,
  `localidade` VARCHAR(45) NULL,
  `password` VARCHAR(45) NOT NULL,
  `username` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`idFunc`),
  UNIQUE INDEX `nif_UNIQUE` (`nif` ASC))
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `habitat`.`Familia`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `habitat`.`Familia` ;

CREATE TABLE IF NOT EXISTS `habitat`.`Familia` (
  `idFam` INT NOT NULL AUTO_INCREMENT,
  `nome` VARCHAR(45) NULL,
  `morada` VARCHAR(45) NULL,
  `telefone` INT NULL,
  `idFunc` INT NOT NULL,
  `nif` INT NULL,
  `dataNascimento` DATE NULL,
  `apelido` VARCHAR(45) NULL,
```

```
  `dataCriaFam` DATE NULL,
  PRIMARY KEY (`idFam`),
  INDEX `fk_Familia_Funcionario1_idx` (`idFunc` ASC),
  UNIQUE INDEX `nif_UNIQUE` (`nif` ASC),
  CONSTRAINT `fk_Familia_Funcionario1`
    FOREIGN KEY (`idFunc`)
    REFERENCES `habitat`.`Funcionario` (`idFunc`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `habitat`.`Candidatura`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `habitat`.`Candidatura` ;

CREATE TABLE IF NOT EXISTS `habitat`.`Candidatura` (
  `idCand` INT NOT NULL AUTO_INCREMENT,
  `idFam` INT NOT NULL,
  `dataCand` DATE NULL,
  `rendimento` DECIMAL(10,2) NULL,
  `estado` ENUM('APROVADO','NAOAPROVADO','NAOACEITE','PENDENTE') NULL DEFAULT 'PENDENTE',
  PRIMARY KEY (`idCand`),
  INDEX `fk_Candidatura_Familia1_idx` (`idFam` ASC),
  CONSTRAINT `fk_Candidatura_Familia1`
    FOREIGN KEY (`idFam`)
    REFERENCES `habitat`.`Familia` (`idFam`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `habitat`.`Projeto`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `habitat`.`Projeto` ;

CREATE TABLE IF NOT EXISTS `habitat`.`Projeto` (
  `idProj` INT NOT NULL AUTO_INCREMENT,
  `orcamento` INT NULL,
  `prazo` DATE NULL,
  `dataInicio` DATE NULL,
  `dataFim` DATE NULL,
  `idFunc` INT NOT NULL,
  `idCand` INT NOT NULL,
  `designacao` VARCHAR(75) NULL,
  `descricao` VARCHAR(200) NULL,
  `dataCriaProj` DATE NULL,
  PRIMARY KEY (`idProj`),
  INDEX `fk_Projeto_Funcionario1_idx` (`idFunc` ASC),
  INDEX `fk_Projeto_Candidatura1_idx` (`idCand` ASC),
  CONSTRAINT `fk_Projeto_Funcionario1`
    FOREIGN KEY (`idFunc`)
    REFERENCES `habitat`.`Funcionario` (`idFunc`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
```

```
  CONSTRAINT `fk_Projeto_Candidatura1`
    FOREIGN KEY (`idCand`)
    REFERENCES `habitat`.`Candidatura` (`idCand`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `habitat`.`Tarefa`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `habitat`.`Tarefa` ;

CREATE TABLE IF NOT EXISTS `habitat`.`Tarefa` (
  `idTar` INT NOT NULL AUTO_INCREMENT,
  `designacao` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`idTar`))
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `habitat`.`TarefaProjeto`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `habitat`.`TarefaProjeto` ;

CREATE TABLE IF NOT EXISTS `habitat`.`TarefaProjeto` (
  `idProj` INT NOT NULL,
  `idTar` INT NOT NULL,
  `dataPlInicio` DATETIME NULL,
  `dataPlFim` DATETIME NULL,
  `dataExInicio` DATETIME NULL,
  `dataExFim` DATETIME NULL,
  PRIMARY KEY (`idProj`, `idTar`),
  INDEX `fk_Projeto_has_Tarefa_Tarefa1_idx` (`idTar` ASC),
  INDEX `fk_Projeto_has_Tarefa_Projeto_idx` (`idProj` ASC),
  CONSTRAINT `fk_Projeto_has_Tarefa_Projeto`
    FOREIGN KEY (`idProj`)
    REFERENCES `habitat`.`Projeto` (`idProj`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Projeto_has_Tarefa_Tarefa1`
    FOREIGN KEY (`idTar`)
    REFERENCES `habitat`.`Tarefa` (`idTar`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `habitat`.`TipoDonativo`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `habitat`.`TipoDonativo` ;

CREATE TABLE IF NOT EXISTS `habitat`.`TipoDonativo` (
  `idTipoDon` INT NOT NULL AUTO_INCREMENT,
  `designacao` VARCHAR(50) NOT NULL,
  `unidade` VARCHAR(20) NOT NULL,
```

```
  PRIMARY KEY (`idTipoDon`))
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `habitat`.`Evento`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `habitat`.`Evento` ;

CREATE TABLE IF NOT EXISTS `habitat`.`Evento` (
  `idEv` INT NOT NULL AUTO_INCREMENT,
  `designacao` VARCHAR(100) NOT NULL,
  `data` DATE NULL,
  `idFunc` INT NOT NULL,
  `descricao` VARCHAR(200) NULL,
  PRIMARY KEY (`idEv`),
  INDEX `fk_Evento_Funcionario1_idx` (`idFunc` ASC),
  CONSTRAINT `fk_Evento_Funcionario1`
    FOREIGN KEY (`idFunc`)
    REFERENCES `habitat`.`Funcionario` (`idFunc`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `habitat`.`Individuo`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `habitat`.`Individuo` ;

CREATE TABLE IF NOT EXISTS `habitat`.`Individuo` (
  `idIndiv` INT NOT NULL AUTO_INCREMENT,
  `idFunc` INT NOT NULL,
  `nome` VARCHAR(75) NOT NULL,
  `dataNascimento` DATE NULL,
  `profissao` VARCHAR(45) NULL,
  `morada` VARCHAR(75) NULL,
  `codigoPostal` VARCHAR(10) NULL,
  `localidade` VARCHAR(60) NULL,
  `telefone` VARCHAR(20) NULL,
  `telemovel` VARCHAR(20) NULL,
  `email` VARCHAR(45) NULL,
  `habilitacoes` VARCHAR(45) NULL,
  `conhecimentosLing` VARCHAR(45) NULL,
  `formacaoComp` VARCHAR(45) NULL,
  `experienciaVolunt` VARCHAR(75) NULL,
  `conhecmentosConstr` VARCHAR(75) NULL,
  `trabalharJuntoVolunt` TINYINT(1) NULL,
  `disponibilidade` VARCHAR(150) NULL,
  `comoConheceu` VARCHAR(75) NULL,
  `receberInfo` TINYINT(1) NULL,
  `isParceiro` TINYINT(1) NOT NULL DEFAULT FALSE,
  `nif` INT NULL,
  `isColectivo` TINYINT(1) NOT NULL DEFAULT FALSE,
  `isDoador` TINYINT(1) NOT NULL DEFAULT FALSE,
  `isVoluntario` TINYINT(1) NOT NULL DEFAULT FALSE,
  `nacionalidadeIndev` VARCHAR(45) NULL,
```

```
  `dataCriaIndiv` DATE NULL,
  PRIMARY KEY (`idIndiv`),
  INDEX `fk_Voluntario_Funcionario1_idx` (`idFunc` ASC),
  UNIQUE INDEX `nif_UNIQUE` (`nif` ASC),
  CONSTRAINT `fk_Voluntario_Funcionario1`
    FOREIGN KEY (`idFunc`)
    REFERENCES `habitat`.`Funcionario` (`idFunc`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `habitat`.`Donativo`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `habitat`.`Donativo` ;

CREATE TABLE IF NOT EXISTS `habitat`.`Donativo` (
  `idDon` INT NOT NULL AUTO_INCREMENT,
  `descricao` VARCHAR(100) NULL,
  `quantInicial` INT NOT NULL,
  `quantAtual` INT UNSIGNED NOT NULL,
  `idTipoDon` INT NOT NULL,
  `dataDon` DATE NULL,
  `idFunc` INT NOT NULL,
  `idEv` INT NOT NULL,
  `idIndiv` INT NOT NULL,
  PRIMARY KEY (`idDon`),
  INDEX `fk_Donativo_TipoDonativo1_idx` (`idTipoDon` ASC),
  INDEX `fk_Donativo_Evento1_idx` (`idEv` ASC),
  INDEX `fk_Donativo_Funcionario1_idx` (`idFunc` ASC),
  INDEX `fk_Donativo_Individuo1_idx` (`idIndiv` ASC),
  CONSTRAINT `fk_Donativo_TipoDonativo1`
    FOREIGN KEY (`idTipoDon`)
    REFERENCES `habitat`.`TipoDonativo` (`idTipoDon`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Donativo_Evento1`
    FOREIGN KEY (`idEv`)
    REFERENCES `habitat`.`Evento` (`idEv`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Donativo_Funcionario1`
    FOREIGN KEY (`idFunc`)
    REFERENCES `habitat`.`Funcionario` (`idFunc`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Donativo_Individuo1`
    FOREIGN KEY (`idIndiv`)
    REFERENCES `habitat`.`Individuo` (`idIndiv`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `habitat`.`Equipa`
```

```
-- -----------------------------------------------------
DROP TABLE IF EXISTS `habitat`.`Equipa` ;

CREATE TABLE IF NOT EXISTS `habitat`.`Equipa` (
  `idEq` INT NOT NULL AUTO_INCREMENT,
  `nacionalidadeEq` VARCHAR(45) NULL,
  `designacao` VARCHAR(100) NOT NULL,
  `dataCriaEq` DATETIME NOT NULL,
  `idFunc` INT NOT NULL,
  PRIMARY KEY (`idEq`),
  INDEX `fk_Equipa_Funcionario1_idx` (`idFunc` ASC),
  CONSTRAINT `fk_Equipa_Funcionario1`
    FOREIGN KEY (`idFunc`)
    REFERENCES `habitat`.`Funcionario` (`idFunc`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `habitat`.`EquipaTarefaProjeto`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `habitat`.`EquipaTarefaProjeto` ;

CREATE TABLE IF NOT EXISTS `habitat`.`EquipaTarefaProjeto` (
  `idEq` INT NOT NULL,
  `dataTrabalho` DATE NOT NULL,
  `duracao` INT NOT NULL,
  `idProj` INT NOT NULL,
  `idTar` INT NOT NULL,
  PRIMARY KEY (`idEq`, `idProj`, `idTar`),
  INDEX `fk_Equipa_has_Projeto_Equipa1_idx` (`idEq` ASC),
  INDEX `fk_EquipaProjeto_TarefaProjeto1_idx` (`idProj` ASC, `idTar` ASC),
  CONSTRAINT `fk_Equipa_has_Projeto_Equipa1`
    FOREIGN KEY (`idEq`)
    REFERENCES `habitat`.`Equipa` (`idEq`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_EquipaProjeto_TarefaProjeto1`
    FOREIGN KEY (`idProj` , `idTar`)
    REFERENCES `habitat`.`TarefaProjeto` (`idProj` , `idTar`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `habitat`.`IndividuoTarefaProjeto`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `habitat`.`IndividuoTarefaProjeto` ;

CREATE TABLE IF NOT EXISTS `habitat`.`IndividuoTarefaProjeto` (
  `idIndiv` INT NOT NULL,
  `dataTrabalho` DATE NOT NULL,
  `duracao` INT NOT NULL,
  `idProj` INT NOT NULL,
  `idTar` INT NOT NULL,
```

```
  PRIMARY KEY (`idIndiv`, `idProj`, `idTar`),
  INDEX `fk_Voluntario_has_Projeto_Voluntario1_idx` (`idIndiv` ASC),
  INDEX `fk_VoluntarioProjeto_TarefaProjeto1_idx` (`idProj` ASC, `idTar` ASC),
  CONSTRAINT `fk_Voluntario_has_Projeto_Voluntario1`
    FOREIGN KEY (`idIndiv`)
    REFERENCES `habitat`.`Individuo` (`idIndiv`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_VoluntarioProjeto_TarefaProjeto1`
    FOREIGN KEY (`idProj` , `idTar`)
    REFERENCES `habitat`.`TarefaProjeto` (`idProj` , `idTar`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `habitat`.`EquipaIndividuo`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `habitat`.`EquipaIndividuo` ;

CREATE TABLE IF NOT EXISTS `habitat`.`EquipaIndividuo` (
  `idIndiv` INT NOT NULL,
  `idEq` INT NOT NULL,
  PRIMARY KEY (`idIndiv`, `idEq`),
  INDEX `fk_Voluntario_has_Equipa_Equipa1_idx` (`idEq` ASC),
  INDEX `fk_Voluntario_has_Equipa_Voluntario1_idx` (`idIndiv` ASC),
  CONSTRAINT `fk_Voluntario_has_Equipa_Voluntario1`
    FOREIGN KEY (`idIndiv`)
    REFERENCES `habitat`.`Individuo` (`idIndiv`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Voluntario_has_Equipa_Equipa1`
    FOREIGN KEY (`idEq`)
    REFERENCES `habitat`.`Equipa` (`idEq`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `habitat`.`Actividade`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `habitat`.`Actividade` ;

CREATE TABLE IF NOT EXISTS `habitat`.`Actividade` (
  `idAtividade` INT NOT NULL AUTO_INCREMENT,
  `designacao` VARCHAR(55) NULL,
  PRIMARY KEY (`idAtividade`))
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `habitat`.`IndividuoActividade`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `habitat`.`IndividuoActividade` ;
```

```
CREATE TABLE IF NOT EXISTS `habitat`.`IndividuoActividade` (
  `idIndiv` INT NOT NULL,
  `idAtividade` INT NOT NULL,
  PRIMARY KEY (`idIndiv`, `idAtividade`),
  INDEX `fk_Voluntario_has_Atividade_Atividade1_idx` (`idAtividade` ASC),
  INDEX `fk_Voluntario_has_Atividade_Voluntario1_idx` (`idIndiv` ASC),
  CONSTRAINT `fk_Voluntario_has_Atividade_Voluntario1`
    FOREIGN KEY (`idIndiv`)
    REFERENCES `habitat`.`Individuo` (`idIndiv`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Voluntario_has_Atividade_Atividade1`
    FOREIGN KEY (`idAtividade`)
    REFERENCES `habitat`.`Actividade` (`idAtividade`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `habitat`.`Questao`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `habitat`.`Questao` ;

CREATE TABLE IF NOT EXISTS `habitat`.`Questao` (
  `idQuestao` INT NOT NULL AUTO_INCREMENT,
  `descricao` VARCHAR(200) NOT NULL,
  `ativa` TINYINT(1) NOT NULL,
  PRIMARY KEY (`idQuestao`))
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `habitat`.`CandidaturaQuestao`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `habitat`.`CandidaturaQuestao` ;

CREATE TABLE IF NOT EXISTS `habitat`.`CandidaturaQuestao` (
  `idCand` INT NOT NULL,
  `idQuestao` INT NOT NULL,
  `resposta` VARCHAR(100) NULL,
  PRIMARY KEY (`idCand`, `idQuestao`),
  INDEX `fk_Candidatura_has_Questao_Questao1_idx` (`idQuestao` ASC),
  INDEX `fk_Candidatura_has_Questao_Candidatura1_idx` (`idCand` ASC),
  CONSTRAINT `fk_Candidatura_has_Questao_Candidatura1`
    FOREIGN KEY (`idCand`)
    REFERENCES `habitat`.`Candidatura` (`idCand`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Candidatura_has_Questao_Questao1`
    FOREIGN KEY (`idQuestao`)
    REFERENCES `habitat`.`Questao` (`idQuestao`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

```
-- -----------------------------------------------------
-- Table `habitat`.`ElementoFamilia`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `habitat`.`ElementoFamilia` ;

CREATE TABLE IF NOT EXISTS `habitat`.`ElementoFamilia` (
  `idElemento` INT NOT NULL AUTO_INCREMENT,
  `idFam` INT NOT NULL,
  `nome` VARCHAR(75) NOT NULL,
  `parentesco` VARCHAR(20) NULL,
  `dataNascimento` DATE NULL,
  `estadoCivil` VARCHAR(20) NULL,
  `ocupacao` VARCHAR(55) NULL,
  `escolaridade` VARCHAR(20) NULL,
  PRIMARY KEY (`idElemento`),
  INDEX `fk_ElementoFamilia_Familia1_idx` (`idFam` ASC),
  CONSTRAINT `fk_ElementoFamilia_Familia1`
    FOREIGN KEY (`idFam`)
    REFERENCES `habitat`.`Familia` (`idFam`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `habitat`.`Prestacao`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `habitat`.`Prestacao` ;

CREATE TABLE IF NOT EXISTS `habitat`.`Prestacao` (
  `idPrestacao` INT NOT NULL AUTO_INCREMENT,
  `idFunc` INT NOT NULL,
  `valor` DECIMAL(10,2) NULL,
  `data` DATE NULL,
  `idFam` INT NOT NULL,
  PRIMARY KEY (`idPrestacao`),
  INDEX `fk_Prestacao_Funcionario1_idx` (`idFunc` ASC),
  INDEX `fk_Prestacao_Familia1_idx` (`idFam` ASC),
  CONSTRAINT `fk_Prestacao_Funcionario1`
    FOREIGN KEY (`idFunc`)
    REFERENCES `habitat`.`Funcionario` (`idFunc`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Prestacao_Familia1`
    FOREIGN KEY (`idFam`)
    REFERENCES `habitat`.`Familia` (`idFam`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `habitat`.`DonativoProjeto`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `habitat`.`DonativoProjeto` ;

CREATE TABLE IF NOT EXISTS `habitat`.`DonativoProjeto` (
```

```
  `idDon` INT NOT NULL,
  `idProj` INT NOT NULL,
  `dataDonProj` DATE NULL,
  `quantDonProj` INT NULL,
  `idFunc` INT NOT NULL,
  PRIMARY KEY (`idDon`, `idProj`),
  INDEX `fk_Donativo_has_Projeto_Projeto1_idx` (`idProj` ASC),
  INDEX `fk_Donativo_has_Projeto_Donativo1_idx` (`idDon` ASC),
  INDEX `fk_DonativoProjeto_Funcionario1_idx` (`idFunc` ASC),
  CONSTRAINT `fk_Donativo_has_Projeto_Donativo1`
    FOREIGN KEY (`idDon`)
    REFERENCES `habitat`.`Donativo` (`idDon`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Donativo_has_Projeto_Projeto1`
    FOREIGN KEY (`idProj`)
    REFERENCES `habitat`.`Projeto` (`idProj`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_DonativoProjeto_Funcionario1`
    FOREIGN KEY (`idFunc`)
    REFERENCES `habitat`.`Funcionario` (`idFunc`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `habitat`.`Relatorio`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `habitat`.`Relatorio` ;

CREATE TABLE IF NOT EXISTS `habitat`.`Relatorio` (
  `idRel` INT NOT NULL AUTO_INCREMENT,
  `obsRel` VARCHAR(45) NULL,
  `dataRel` DATE NULL,
  `ficheiroRel` BLOB NULL,
  `idFam` INT NOT NULL,
  `idFunc` INT NOT NULL,
  PRIMARY KEY (`idRel`),
  INDEX `fk_Relatorio_Familia1_idx` (`idFam` ASC),
  INDEX `fk_Relatorio_Funcionario1_idx` (`idFunc` ASC),
  CONSTRAINT `fk_Relatorio_Familia1`
    FOREIGN KEY (`idFam`)
    REFERENCES `habitat`.`Familia` (`idFam`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Relatorio_Funcionario1`
    FOREIGN KEY (`idFunc`)
    REFERENCES `habitat`.`Funcionario` (`idFunc`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;

USE `habitat` ;

-- -----------------------------------------------------
```

```
-- Placeholder table for view `habitat`.`ViewTarefaProjeto`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `habitat`.`ViewTarefaProjeto` (`idProj` INT, `idTar` INT, `
designacaoTar` INT, `dataPlInicio` INT, `dataPlFim` INT, `DataExInicio` INT, `DataExFim` INT);

-- -----------------------------------------------------
-- Placeholder table for view `habitat`.`ViewCandidaturaQuestao`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `habitat`.`ViewCandidaturaQuestao` (`idCand` INT, `idQuestao` INT,
 `resposta` INT, `questao` INT, `ativa` INT);

-- -----------------------------------------------------
-- View `habitat`.`ViewTarefaProjeto`
-- -----------------------------------------------------
DROP VIEW IF EXISTS `habitat`.`ViewTarefaProjeto` ;
DROP TABLE IF EXISTS `habitat`.`ViewTarefaProjeto`;
USE `habitat`;
CREATE  OR REPLACE VIEW `ViewTarefaProjeto` AS
Select idProj, t.idTar as idTar, t.designacao as designacaoTar, dataPlInicio, dataPlFim,
DataExInicio, DataExFim from TarefaProjeto tp inner join Tarefa t on t.idtar = tp.idtar;

-- -----------------------------------------------------
-- View `habitat`.`ViewCandidaturaQuestao`
-- -----------------------------------------------------
DROP VIEW IF EXISTS `habitat`.`ViewCandidaturaQuestao` ;
DROP TABLE IF EXISTS `habitat`.`ViewCandidaturaQuestao`;
USE `habitat`;
CREATE  OR REPLACE VIEW `ViewCandidaturaQuestao` AS
select `c`.`idCand` AS `idCand`,`c`.`idQuestao` AS `idQuestao`,`c`.`resposta` AS `resposta`,`
q`.`descricao` AS `questao`,`q`.`ativa` AS `ativa` from (`CandidaturaQuestao` `c` join `
Questao` `q` on((`q`.`idQuestao` = `c`.`idQuestao`)))
;
USE `habitat`;

DELIMITER $$

USE `habitat`$$
DROP TRIGGER IF EXISTS `habitat`.`TirarStock` $$
USE `habitat`$$
CREATE TRIGGER `TirarStock` BEFORE INSERT ON `DonativoProjeto`
FOR EACH ROW
BEGIN
    UPDATE Donativo Set quantAtual = quantAtual - NEW.quantDonProj where idDon = NEW.idDon;
    if (@@error_count > 0) then
        SIGNAL SQLSTATE '45000'
            SET MESSAGE_TEXT = 'Quantidade Actual inferior a 0', MYSQL_ERRNO = 1001;
    end if;
END$$



USE `habitat`$$
DROP TRIGGER IF EXISTS `habitat`.`TirarSockUpdate` $$
USE `habitat`$$
CREATE TRIGGER `TirarSockUpdate` BEFORE UPDATE ON `DonativoProjeto`
FOR EACH ROW
BEGIN
    UPDATE Donativo Set quantAtual = quantAtual + OLD.quantDonProj - NEW.quantDonProj where
```

```
    idDon = NEW.idDon;
    if (@@error_count > 0) then
        SIGNAL SQLSTATE '45000'
            SET MESSAGE_TEXT = 'Quantidade Actual inferior a 0', MYSQL_ERRNO = 1001;
    end if;
END$$


USE `habitat`$$
DROP TRIGGER IF EXISTS `habitat`.`ReporStock` $$
USE `habitat`$$
CREATE TRIGGER `ReporStock` BEFORE DELETE ON `DonativoProjeto`
FOR EACH ROW
BEGIN
    UPDATE Donativo Set quantAtual = quantAtual + OLD.quantDonProj where idDon = OLD.idDon;
END$$


DELIMITER ;
SET SQL_MODE = '';
GRANT USAGE ON *.* TO admin;
 DROP USER admin;
SET SQL_MODE='TRADITIONAL,ALLOW_INVALID_DATES';
CREATE USER 'admin' IDENTIFIED BY 'admin';

GRANT ALL ON TABLE `habitat`.`CandidaturaQuestao` TO 'admin';
GRANT ALL ON TABLE `habitat`.`Prestacao` TO 'admin';
GRANT ALL ON TABLE `habitat`.`Relatorio` TO 'admin';
GRANT ALL ON TABLE `habitat`.`ElementoFamilia` TO 'admin';
GRANT ALL ON TABLE `habitat`.`Familia` TO 'admin';
GRANT ALL ON TABLE `habitat`.`EquipaTarefaProjeto` TO 'admin';
GRANT ALL ON TABLE `habitat`.`IndividuoTarefaProjeto` TO 'admin';
GRANT ALL ON TABLE `habitat`.`TarefaProjeto` TO 'admin';
GRANT ALL ON TABLE `habitat`.`ViewTarefaProjeto` TO 'admin';
GRANT ALL ON TABLE `habitat`.`Projeto` TO 'admin';
GRANT ALL ON TABLE `habitat`.`DonativoProjeto` TO 'admin';
GRANT ALL ON TABLE `habitat`.`Donativo` TO 'admin';
GRANT ALL ON TABLE `habitat`.`Individuo` TO 'admin';
GRANT ALL ON TABLE `habitat`.`EquipaIndividuo` TO 'admin';
GRANT ALL ON TABLE `habitat`.`Evento` TO 'admin';
GRANT ALL ON TABLE `habitat`.`IndividuoActividade` TO 'admin';
GRANT ALL ON TABLE `habitat`.`Equipa` TO 'admin';
GRANT ALL ON TABLE `habitat`.`Candidatura` TO 'admin';
GRANT ALL ON TABLE `habitat`.`TipoDonativo` TO 'admin';
GRANT ALL ON TABLE `habitat`.`Funcionario` TO 'admin';
GRANT ALL ON TABLE `habitat`.`Actividade` TO 'admin';
GRANT ALL ON TABLE `habitat`.`Tarefa` TO 'admin';
GRANT ALL ON TABLE `habitat`.`Questao` TO 'admin';
SET SQL_MODE = '';
GRANT USAGE ON *.* TO fundos;
 DROP USER fundos;
SET SQL_MODE='TRADITIONAL,ALLOW_INVALID_DATES';
CREATE USER 'fundos' IDENTIFIED BY 'fundos';

GRANT DELETE, INSERT, SELECT, UPDATE ON TABLE `habitat`.`DonativoProjeto` TO 'fundos';
GRANT DELETE, INSERT, SELECT, UPDATE ON TABLE `habitat`.`Donativo` TO 'fundos';
GRANT DELETE, INSERT, SELECT, UPDATE ON TABLE `habitat`.`Individuo` TO 'fundos';
```

```
GRANT DELETE, INSERT, SELECT, UPDATE ON TABLE `habitat`.`EquipaIndividuo` TO 'fundos';
GRANT DELETE, INSERT, SELECT, UPDATE ON TABLE `habitat`.`Evento` TO 'fundos';
GRANT DELETE, INSERT, SELECT, UPDATE ON TABLE `habitat`.`IndividuoActividade` TO 'fundos';
GRANT DELETE, INSERT, SELECT, UPDATE ON TABLE `habitat`.`Equipa` TO 'fundos';
GRANT SELECT ON TABLE `habitat`.`EquipaTarefaProjeto` TO 'fundos';
GRANT SELECT ON TABLE `habitat`.`IndividuoTarefaProjeto` TO 'fundos';
GRANT SELECT ON TABLE `habitat`.`TarefaProjeto` TO 'fundos';
GRANT SELECT ON TABLE `habitat`.`ViewTarefaProjeto` TO 'fundos';
GRANT SELECT ON TABLE `habitat`.`Projeto` TO 'fundos';
GRANT SELECT ON TABLE `habitat`.`DonativoProjeto` TO 'fundos';
GRANT SELECT ON TABLE `habitat`.`Donativo` TO 'fundos';
GRANT SELECT ON TABLE `habitat`.`Individuo` TO 'fundos';
GRANT SELECT ON TABLE `habitat`.`EquipaIndividuo` TO 'fundos';
GRANT SELECT ON TABLE `habitat`.`Evento` TO 'fundos';
GRANT SELECT ON TABLE `habitat`.`IndividuoActividade` TO 'fundos';
GRANT SELECT ON TABLE `habitat`.`Equipa` TO 'fundos';
GRANT SELECT ON TABLE `habitat`.`CandidaturaQuestao` TO 'fundos';
GRANT SELECT ON TABLE `habitat`.`Relatorio` TO 'fundos';
GRANT SELECT ON TABLE `habitat`.`Candidatura` TO 'fundos';
GRANT SELECT ON TABLE `habitat`.`ElementoFamilia` TO 'fundos';
GRANT SELECT ON TABLE `habitat`.`Familia` TO 'fundos';
GRANT SELECT ON TABLE `habitat`.`Prestacao` TO 'fundos';
SET SQL_MODE = '';
GRANT USAGE ON *.* TO obras;
 DROP USER obras;
SET SQL_MODE='TRADITIONAL,ALLOW_INVALID_DATES';
CREATE USER 'obras' IDENTIFIED BY 'obras';

GRANT DELETE, INSERT, SELECT, UPDATE ON TABLE `habitat`.`EquipaTarefaProjeto` TO 'obras';
GRANT DELETE, INSERT, SELECT, UPDATE ON TABLE `habitat`.`IndividuoTarefaProjeto` TO 'obras';
GRANT DELETE, INSERT, SELECT, UPDATE ON TABLE `habitat`.`TarefaProjeto` TO 'obras';
GRANT DELETE, INSERT, SELECT, UPDATE ON TABLE `habitat`.`ViewTarefaProjeto` TO 'obras';
GRANT DELETE, INSERT, SELECT, UPDATE ON TABLE `habitat`.`Projeto` TO 'obras';
GRANT INSERT, DELETE, SELECT, UPDATE ON TABLE `habitat`.`DonativoProjeto` TO 'obras';
GRANT SELECT ON TABLE `habitat`.`EquipaTarefaProjeto` TO 'obras';
GRANT SELECT ON TABLE `habitat`.`IndividuoTarefaProjeto` TO 'obras';
GRANT SELECT ON TABLE `habitat`.`TarefaProjeto` TO 'obras';
GRANT SELECT ON TABLE `habitat`.`ViewTarefaProjeto` TO 'obras';
GRANT SELECT ON TABLE `habitat`.`Projeto` TO 'obras';
GRANT SELECT ON TABLE `habitat`.`DonativoProjeto` TO 'obras';
GRANT SELECT ON TABLE `habitat`.`Donativo` TO 'obras';
GRANT SELECT ON TABLE `habitat`.`Individuo` TO 'obras';
GRANT SELECT ON TABLE `habitat`.`EquipaIndividuo` TO 'obras';
GRANT SELECT ON TABLE `habitat`.`Evento` TO 'obras';
GRANT SELECT ON TABLE `habitat`.`IndividuoActividade` TO 'obras';
GRANT SELECT ON TABLE `habitat`.`Equipa` TO 'obras';
GRANT SELECT ON TABLE `habitat`.`CandidaturaQuestao` TO 'obras';
GRANT SELECT ON TABLE `habitat`.`Relatorio` TO 'obras';
GRANT SELECT ON TABLE `habitat`.`Candidatura` TO 'obras';
GRANT SELECT ON TABLE `habitat`.`ElementoFamilia` TO 'obras';
GRANT SELECT ON TABLE `habitat`.`Familia` TO 'obras';
GRANT SELECT ON TABLE `habitat`.`Prestacao` TO 'obras';
SET SQL_MODE = '';
GRANT USAGE ON *.* TO fundos;
 DROP USER fundos;
SET SQL_MODE='TRADITIONAL,ALLOW_INVALID_DATES';
CREATE USER 'fundos' IDENTIFIED BY 'fundos';
```

```
GRANT DELETE, INSERT, SELECT, UPDATE ON TABLE `habitat`.`DonativoProjeto` TO 'fundos';
GRANT DELETE, INSERT, SELECT, UPDATE ON TABLE `habitat`.`Donativo` TO 'fundos';
GRANT DELETE, INSERT, SELECT, UPDATE ON TABLE `habitat`.`Individuo` TO 'fundos';
GRANT DELETE, INSERT, SELECT, UPDATE ON TABLE `habitat`.`EquipaIndividuo` TO 'fundos';
GRANT DELETE, INSERT, SELECT, UPDATE ON TABLE `habitat`.`Evento` TO 'fundos';
GRANT DELETE, INSERT, SELECT, UPDATE ON TABLE `habitat`.`IndividuoActividade` TO 'fundos';
GRANT DELETE, INSERT, SELECT, UPDATE ON TABLE `habitat`.`Equipa` TO 'fundos';
GRANT SELECT ON TABLE `habitat`.`EquipaTarefaProjeto` TO 'fundos';
GRANT SELECT ON TABLE `habitat`.`IndividuoTarefaProjeto` TO 'fundos';
GRANT SELECT ON TABLE `habitat`.`TarefaProjeto` TO 'fundos';
GRANT SELECT ON TABLE `habitat`.`ViewTarefaProjeto` TO 'fundos';
GRANT SELECT ON TABLE `habitat`.`Projeto` TO 'fundos';
GRANT SELECT ON TABLE `habitat`.`DonativoProjeto` TO 'fundos';
GRANT SELECT ON TABLE `habitat`.`Donativo` TO 'fundos';
GRANT SELECT ON TABLE `habitat`.`Individuo` TO 'fundos';
GRANT SELECT ON TABLE `habitat`.`EquipaIndividuo` TO 'fundos';
GRANT SELECT ON TABLE `habitat`.`Evento` TO 'fundos';
GRANT SELECT ON TABLE `habitat`.`IndividuoActividade` TO 'fundos';
GRANT SELECT ON TABLE `habitat`.`Equipa` TO 'fundos';
GRANT SELECT ON TABLE `habitat`.`CandidaturaQuestao` TO 'fundos';
GRANT SELECT ON TABLE `habitat`.`Relatorio` TO 'fundos';
GRANT SELECT ON TABLE `habitat`.`Candidatura` TO 'fundos';
GRANT SELECT ON TABLE `habitat`.`ElementoFamilia` TO 'fundos';
GRANT SELECT ON TABLE `habitat`.`Familia` TO 'fundos';
GRANT SELECT ON TABLE `habitat`.`Prestacao` TO 'fundos';
SET SQL_MODE = '';
GRANT USAGE ON *.* TO guest;
 DROP USER guest;
SET SQL_MODE='TRADITIONAL,ALLOW_INVALID_DATES';
CREATE USER 'guest';

SET SQL_MODE = '';
GRANT USAGE ON *.* TO loginCheck;
 DROP USER loginCheck;
SET SQL_MODE='TRADITIONAL,ALLOW_INVALID_DATES';
CREATE USER 'loginCheck' IDENTIFIED BY 'tpV8jKd9YwSzLvQfzHLUZz9Jua3mt5VnTFWvsVczTNanJxSh';

GRANT SELECT ON TABLE `habitat`.`Funcionario` TO 'loginCheck';

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

VI. Anexo – Script de Povoamento da BD

```
INSERT INTO Funcionario (idFunc,tipoFunc,nome,nif,morada,localidade,telefone,telemovel,
codigoPostal,dataNascimento,username,password)
VALUES (1,"ADMIN", "Rui Oliveira", 1234123, "R. 5 de Outubro","Oliveira Santa Maria",
"+351971234234", "+351234432765", "5434-343","1994-11-10", "User", "password"),
    (2,"FAM", "Rufino Felipe", 983423423, "R. André Cunha","Valadares", "+351973234754",
    "+351234834835", "4762-012","1991-02-23", "Rufinho", "indecifravel"),
    (3,"Fundos", "Teresa Frederica", 965475632, "R. da Minha","Vilar do Paraiso",
    "+351976524985", "+351234785412", "4712-012","1993-02-02", "Teresa", "teresa"),
    (4,"Obras", "Antonio Mafalda", 652365789, "R. Casais", "Gulpilhares", "+351976589210",
    "+351234985478", "4564-312","1993-09-15", "Antonio", "tone");

INSERT INTO Tarefa (idTar, designacao) VALUES
(1, "Colar Papel Interior"),
(2, "Pintar exterior"),
(3, "Colocar soalho"),
(4, "Instalação de tubagens"),
(5, "Limpar exterior"),
(6, "Instalação de pavimentos"),
(7, "Instalar quadro eletrico");

INSERT INTO Actividade (idAtividade, designacao) VALUES
(1, "Contrução civil"),
(2, "Pintura"),
(3, "Eletrotécnica"),
(4, "Carpintaria"),
(5, "Serviços");

INSERT INTO Questao (idQuestao, descricao, ativa) VALUES
(1, "Esta disposto a trabalhar conosco?",true),
(2, "Quais os problemas da casa atual?",true),
(3, "Tem terreno?",true),
(4, "Durante a obra, terá onde ficar?",true),
(5, "Em que dias estaria disponivel para trabalhar na obra?",true);

INSERT INTO TipoDonativo (idTipoDon, designacao, unidade) VALUES
(1, "Modetario", "€"),
(2, "Cimento", "sacos"),
(3, "Tijolo", "Kg"),
(4, "Areia", "Kg"),
(5, "Telha", "m2");

INSERT INTO `Individuo` (`idIndiv`,`idFunc`,`nome`,`dataNascimento`,`profissao`,`codigoPostal
`,`localidade`,`telefone`,`email`,`nif`, `isDoador`, `isVoluntario`) VALUES
 (1,"3","Suki Swanson","1977-10-16","Pintor","4234-012","Murcia","0279487496",
 "dis@pedeultrices.ca","952432066", true, false)
,(2,"3","Germane Gilbert","1969-10-09","Eletricista","4985-065","Ronciglione","0548532565",
"orci.Phasellus@tellus.edu","019358906", false, true)
,(3,"3","Basil P. Barry","1963-10-27","Contrutor civil","4874-065","Tramutola","0198111084",
"ullamcorper.viverra@et.ca","861537775", true, true)
,(4,"3","Heather Pierce","1968-10-16","Contrutor civil","4210-125","Piana degli Albanesi",
"0727621183","ipsum.dolor.sit@magna.org","195608919", false, true),
(5,"3","Edu Severina","1988-04-20","Pintor","4652-125","Penacova","9865214578",
"email@gmail.pt","857524541", false, true);

INSERT INTO `Familia` (`idFam`,`nome`,`morada`,`telefone`,`nif`,`dataNascimento`,`apelido`,`
dataCriaFam`,`idFunc`) VALUES
(1,"Tamara Brenna","927-7843 Rutrum Road","08690220","9267378","1976-09-18","Brenna",
```

```
"2009-06-06",2)
,(2,"Gregory Karen","Ap #767-1046 A, St.","03263331","03057537","1980-09-01","Karen",
"2008-10-05",2)
,(3,"Nichole Roanna","Ap #675-8686 Nunc Ave","06494977","359833","1977-01-13","Roanna",
"2007-12-11",2)
,(4,"Yeo Sigourney","838-8439 Nunc Rd.","09179132","9430232","1975-05-19","Sigourney",
"2007-03-17",2)
,(5,"Adelaide Francisco","Ap #945-2966 Porttitor Avenue","06562033","2673","1980-09-29",
"Francisco","2011-08-05",2)
,(6,"Jônatas Tristão","Ap #698-6871 Non, St.","04067542","41511670","1990-10-07","Tristão",
"2011-07-30",2);

INSERT INTO `ElementoFamilia` (`idElemento`,`idFam`,`nome`,`parentesco`,`dataNascimento`,`
estadoCivil`,`ocupacao`,`escolaridade`)
VALUES (1,1,"Lana Crosby","subrinho","1987-04-02","solteiro","Informatica","2ciclo"),
(2,1,"Quincy Cannon","pai","1993-11-04","casado","Motorista","3ciclo"),
(3,2,"Josephine Estrada","pai","1977-04-25","casado","Construtor Civil","ens. Superiror"),
(4,2,"Sybill Velazquez","primo","1981-09-23","solteiro","Agricultor","ens. Superiror"),
(5,3,"Patience Moody","primo","1981-12-01","solteiro","desempregado","2ciclo"),
(6,4,"Amaya Potter","filho","1982-07-10","solteiro","desempregado","ens. Superiror"),
(7,4,"Hyacinth Holden","mae","1976-02-28","solteiro","desempregado","ens. Superiror"),
(8,4,"Mia Sloan","primo","1990-09-27","solteiro","Agricultor","ens. Superiror"),
(9,5,"Ainsley Yang","filho","1989-01-12","casado","desempregado","2ciclo"),
(10,5,"Wendy Fleming","sobrinho","1982-01-28","solteiro","Sapateiro","ens. Superiror"),
(11,5,"Anjolie Martin","sobrinho","1980-02-06","solteiro","Empregado Textil","ens. Superiror"
),
(12,6,"Carolyn Chase","pai","1964-09-06","casado","desempregado","ens. Superiror"),
(13,6,"Katelyn Blankenship","irmao","1955-08-04","casado","Eletricista","3ciclo"),
(14,6,"Indira Savage","irmao","1955-12-30","casado","Empregado Mesa","2ciclo"),
(15,6,"Brynn Pearson","mae","1961-02-09","casado","desempregado","2ciclo");

INSERT INTO `Candidatura` (`idCand`,`idFam`,`dataCand`,`rendimento`, `estado`)
VALUES (1,"1","2014-05-11",2000, 'Aprovado'),
(2,"1","2012-08-10",720, 'NAOAPROVADO'),
(3,"2","2013-12-24",1200, 'Aprovado'),
(4,"3","2013-12-28",776, 'Aprovado'),
(5,"4","2014-09-28",744, 'Aprovado'),
(6,"5","2015-11-19",888, 'Pendente'),
(7,"6","2014-12-29",422, 'Pendente');

INSERT INTO `CandidaturaQuestao` (idCand,idQuestao, resposta)
VALUES (1 ,1,"Nao"),
(1 ,2,"Telhado destruido"),
(1 ,3,"Sim"),
(1 ,4,"Sim"),
(1 ,5,"Nunca"),
(2 ,1,"Sim"),
(2 ,2,"Idade avançada"),
(2 ,3,"Nao"),
(2 ,4,"Sim"),
(2 ,5,"Segundas e Terças"),
(3 ,1,"Nao"),
(3 ,2,"Sem eletricidade"),
(3 ,3,"Sim"),
(3 ,4,"Sim"),
(3 ,5,"Nunca"),
(4 ,1,"Sim"),
```

```
(4 ,2,"Canalização destruida");

INSERT INTO `Projeto` (idProj, orcamento, prazo, dataInicio, dataFim, idFunc, idCand,
designacao, descricao, dataCriaProj) VALUES
(1, 123000.00, "2016-06-29", "2015-07-15", "2016-06-29", 4,1, "Reparação do telhado",
"Reparar o telhado para evitar a chuva","2014-12-21"),
(2, 136000.00, "2015-01-30", "2014-02-14", "2015-01-30", 4,3, "Reparações eletricas",
"Reconstruir a instalação eletrica","2014-12-21"),
(3, 145000.00, "2016-01-05", "2015-01-20", "2016-01-05", 4,4, "Reparações da canalização",
"Reconstruir a canalização","2014-12-21"),
(4, 154000.00, "2016-03-26", "2015-04-11", "2016-03-26", 4,5, "Contrução de casa nova",
"Contruir casa de raiz para familia desalojada","2014-12-21");

INSERT INTO TarefaProjeto (idProj, IdTar, dataPlInicio, DataPlFim)
VALUES(1, 1 ,"2014-10-23","2014-10-15"),
(1, 2 ,"2014-10-12","2014-10-15"),
(2, 3 ,"2014-11-10","2014-10-15"),
(2, 4 ,"2014-10-19","2014-10-15"),
(2, 5 ,"2014-10-20","2014-10-15"),
(3, 1 ,"2014-03-09","2014-02-11"),
(3, 2 ,"2014-03-15","2014-02-11"),
(4, 3 ,"2014-02-07","2014-02-11"),
(4, 4 ,"2014-02-09","2014-02-11"),
(4, 5 ,"2014-03-01","2014-02-11");

INSERT INTO Evento(idEv, designacao, data, idFunc, descricao) VALUES
(1, "Evento espontâneo", DATE(NOW()), 1, "Evento usado para donativos gerais"),
(2, "Angariação de fundos do sr. Luis", "2014-02-23", 1, "Evento usado para donativos de
materiais"),
(3, "Dia de Natal", "2013-12-20", 1, "Evento usado para donativos monetarios");

INSERT INTO IndividuoTarefaProjeto (idIndiv, dataTrabalho, duracao, idProj, idTar)
VALUES(2, "2014-03-12", 12, 1, 1),
(2, "2014-03-23", 24, 3, 2),
(3, "2014-02-10", 43, 2, 3),
(5, "2014-02-13", 55, 2, 4),
(5, "2014-03-04", 4, 2, 5),
(3, "2014-03-18", 46, 3, 1),
(5, "2014-03-15", 45, 1, 2),
(2, "2014-02-14", 15, 4, 3),
(2, "2014-02-11", 14, 4, 4),
(3, "2014-03-06", 7, 4, 5);

INSERT INTO IndividuoActividade(idIndiv, idAtividade)
VALUES(1, 4),
(2, 1),
(3, 2),
(4, 2),
(5, 4),
(1, 2),
(5, 2);

INSERT INTO Equipa(idEq, nacionalidadeEq, designacao, dataCriaEq, idFunc) VALUES
(1, "Portuguesa", "Empresa UnknownX", NOW(), 3),
(2, "Italiana", "Ferias de voluntariado", NOW(), 3),
(3, "Inglesa", "Associação Vicamonesis", NOW(), 3);
```

```
INSERT INTO EquipaIndividuo(idIndiv, idEq)
VALUES(2,1),
(4,1),
(3,2),
(5,3);

INSERT INTO EquipaTarefaProjeto(idEq, dataTrabalho, duracao, idProj, idTar) VALUES
(1 ,"2014-03-15", 5 , 1, 2),
(2 ,"2014-03-02", 2 , 2, 4),
(3 ,"2014-03-22", 12 , 2, 5);

INSERT INTO Donativo(idDon, descricao, quantInicial, quantAtual, idTipoDon, dataDon, idFunc,
idEv, idIndiv)
VALUES(1,'Doação regular',10,10,1,'2014-09-23',3,1,1),
(2,'Doação regular',10,10,1,'2014-10-24',3,1,1),
(3,'Doação de material em excesso',10,10,3,'2014-11-26',3,1,3),
(5,'Doação Monetaria',10,10,1,'2014-09-23',3,1,1);

INSERT INTO DonativoProjeto(idDon, idProj, dataDonProj, quantDonProj, idFunc)
VALUES
(1,1,'2014-10-29',4,3),
(3,2,'2014-11-12',1,3),
(5,2,'2014-10-29',4,3);

INSERT INTO Relatorio(obsRel, dataRel, idFam, idFunc)
VALUES
("Familia a habituar-se normalmente", "2014-12-21",1,1),
("Falta de limpeza externa da casa", "2014-12-21",2,1);

INSERT INTO Prestacao(idFunc, valor, data, idFam)
VALUES
(2,100,"2012-10-21",1),
(2,120,"2014-12-21",1),
(2,130,"2014-12-21",2),
(2,120,"2014-12-21",3),
(2,100,"2014-12-21",4);
```