Modelação de uma Base de Dados

5410 - Bases de Dados - Conceitos

Técnico/a Especialista em Tecnologias e Programação de Sistemas de Informação - TPSI PL 1223

Autores:
Daniel Quaresma
Lucas Silvestre
João Correia
Vladimiro Bonaparte

Formador:
Vitor Custódio

Palmela, 20 de maio de 2024

Conteúdo

1 | Introdução

Foi-nos proposto trabalhar em grupo para idealizar uma aplicação e realizar a modelação de uma base de dados que a suporte.

O primeiro passo consistiu em definir o objetivo e as principais funcionalidades da aplicação. Esta etapa é crucial para o desenvolvimento, pois estabelece a base para a modelação da base de dados.

Após a definição da aplicação, o desafio seguinte foi modelar a base de dados de modo a suportar todas as funcionalidades identificadas. Utilizámos a ferramenta *MySQL Workbench* para desenhar o esquema da base de dados. Este processo envolveu a criação de tabelas, a definição de chaves primárias e estrangeiras, atributos e relacionamentos.

Com as funcionalidades da aplicação claramente definidas e o modelo de dados estabelecido, passámos a criar as queries necessárias para consulta, inserção, atualização e remoção de dados.

Finalmente, com a base de dados modelada e as queries prontas, criámos uma nova base de dados seguindo o modelo desenvolvido e introduzimos dados simulados. Isto permitiu-nos testar as queries e assegurar que o nosso modelo de dados funcionava corretamente.

Figura 1.1: Ferramentas utilizadas

2 | Descrição da aplicação

2.1 | Descrição geral

2.1.1 | Objectivo

Desenvolver uma aplicação que funcione como uma plataforma de mercado, facilitando a divulgação, a compra e venda de ideias e produtos sustentáveis. A aplicação visa principalmente apoiar pequenos agricultores e comércios regionais, oferecendo uma solução eficaz para a promoção e comercialização de cabazes de produtos regionais e outros bens sustentáveis em Portugal.

2.1.2 | Problema a resolver

A aplicação proposta procura enfrentar a dificuldade e a limitada exposição que pequenas iniciativas sustentáveis enfrentam no território português. Pequenos produtores e comerciantes muitas vezes não possuem os recursos ou a visibilidade necessária para atingir um público mais amplo. A falta de exposição restringe suas oportunidades de negócio e crescimento.

2.1.3 | Funcionalidades e benefícios

A plataforma permitirá que os utilizadores tanto vendam quanto comprem ideias e produtos sustentáveis, oferecendo uma exposição virtual para pequenas empresas e particulares que procuram ampliar a sua visibilidade e quota de mercado.

Através desta aplicação, esperamos criar um ecossistema de comércio sustentável que beneficie tanto os vendedores, proporcionando-lhes uma nova via de negócios, quanto os compradores, que terão acesso facilitado a produtos sustentáveis de qualidade.

Figura 2.1: Cabaz de produtos sustentáveis

2.2 | Principais funcionalidades

Optamos por organizar as funcionalidades da nossa aplicação em quatro categorias. Abaixo, apresentamos cada uma, com alguns exemplos específicos de funcionalidades:

- **Users**

  Nesta categoria incluímos todas as funcionalidades relacionadas com utilizadores (não vendedores), tais como:

  - **Registo de utilizador**

    Um novo utilizador deve poder registar-se com os seus dados na nossa aplicação.

  - **Alterar dados de perfil**

    Um utilizador registado deve poder fazer a alteração dos seus dados de perfil.

  - **Criar uma morada de entrega**

    Um utilizador deve poder criar uma morada de entrega para a sua conta.

  - **Apagar morada**

    Um utilizador registado deve poder apagar as suas moradas de entrega.

  - **Apagar registo**

    Um utilizador registado deve poder apagar o seu registo.

- **Products**

  Nesta categoria incluímos todo o tipo de funcionalidades que envolvem os produtos:

  - **Adicionar produto**

    Um vendedor pode adicionar um produto novo para pôr á venda na sua loja.

  - **Adicionar ao carrinho**

    Um utilizador pode adicionar um produto que esteja á venda numa loja ao seu carrinho de compras.

  - **Criar categoria**

    Um vendedor pode adicionar uma nova categoria para os seus produtos.

  - **Adicionar imagem**

    Um vendedor pode adicionar uma imagem a uma galeria de imagens de um determinado produto.

  - **Limpar o carrinho**

    Um utilizador pode remover todos os produtos que adicionou ao seu carrinho de compras de uma só vez.

- **Store**

  Nesta categoria incluímos todo o tipo de funcionalidades que envolvem as lojas:

  - **Registo como vendedor**

    Um utilizador normal deve poder registar-se como vendedor.

  - **Criar nova loja**

    Um vendedor pode criar uma nova loja.

  - **Adicionar imagem**

    Um vendedor pode adicionar uma imagem a uma galeria de imagens de uma determinada loja que lhe pertença.

  - **Eliminar loja**

    Um vendedor deve poder apagar uma loja que lhe pertença.

  - **Modificar avaliaçao da loja**

    Um utilizador deve poder modificar uma avaliação que fez no passado de uma loja que seja cliente.

- **Orders**

  Nesta categoria incluímos todo o tipo de funcionalidades que envolvem as encomendas:

  - **Fazer encomenda**

    Um utilizador deve poder realizar uma encomenda dos produtos que tem no carrinho de compras.

  - **Consultar estado da encomenda**

    Um utilizador pode consultar o estado de uma encomenda que tenha feito.

  - **Cancelar encomenda**

    Um utilizador que tenha feito uma encomenda pode cancelá-la a qualquer momento desde que a encomenda não tenha ainda sido enviada.

  - **Visualizar valor total**

    Um utilizador deve poder visualizar o valor total de uma encomenda que tenha realizado.

  - **Ordenar encomendas**

    Um utilizador deve poder organizar todas as encomendas que realizou por data ou estado.

3 | Modelação da base de dados

3.1 | Diagramas, tabelas e modelos

Com as funcionalidades da aplicação definidas o próximo passo foi modelar a base de dados criando as tabelas e desenhando o diagrama de Entidade-Relacionamento Estendido(EER) através da ferramenta *MySQL Workbench*.

3.1.1 | Tabelas

Criámos as tabelas tendo em conta as funcionalidades da aplicação:

- **user:** Dados dos utilizadores registados na aplicação.

Nome	Tipo	Atributos	Padrão / Expressão
user_id	INT	PK - NN - AI	
vendor_id	INT	FK	NULL
first_name	VARCHAR(100)	NN	
last_name	VARCHAR(100)	NN	
email	VARCHAR(100)	NN - UQ	
deleted	TINYINT	NN	0

- **home_address:** Moradas de entrega associadas aos utilizadores.

Nome	Tipo	Atributos	Padrão / Expressão
home_address_id	INT	PK - NN - AI	
user_id	INT	NN - FK	
first_name	VARCHAR(100)	NN	
last_name	VARCHAR(100)	NN	
phone_number	CHAR(9)	NN - UQ	
street_address	VARCHAR(100)	NN	
postal_code	CHAR(8)	NN	
city	VARCHAR(100)	NN	
comment	MEDIUMTEXT		
deleted	TINYINT	NN	0

- **product:** Produtos inseridos pelos vendedores.

Nome	Tipo	Atributos	Padrão / Expressão
product_id	INT	PK - NN - AI	
product_name	VARCHAR(100)	NN	
description	MEDIUMTEXT	NN	
discount	DOUBLE	NN - UQ	0.0
image_link	VARCHAR(250)		
price	DOUBLE	NN - UQ	
stock	INT	NN - UQ	
is_unit	TINYINT	NN - UQ	0

- **category:** Categorias dos produtos.

Nome	Tipo	Atributos	Padrão / Expressão
category_id	INT	PK - NN - AI	
category_name	VARCHAR(100)	NN	

- **product_gallery:** Galeria de imagens dos produtos.

Nome	Tipo	Atributos	Padrão / Expressão
product_gallery_id	INT	PK - NN - AI	
product_id	INT	NN - FK	
image_link	VARCHAR(250)	NN	

- **product_category:** Ligação entre produtos e categorias.

Nome	Tipo	Atributos	Padrão / Expressão
product_id	INT	PK - NN - FK	
category_id	INT	PK - NN - FK	

- **product_review:** Avaliações dos produtos.

Nome	Tipo	Atributos	Padrão / Expressão
product_review_id	INT	PK - NN - AI	
user_id	INT	NN - FK	
product_id	INT	NN - FK	
rating	INT	NN	
created	TIMESTAMP		CURRENT_TIMESTAMP
comment	MEDIUMTEXT	NN	

- **store:** Lojas registadas pelos vendedores.

Nome	Tipo	Atributos	Padrão / Expressão
store_id	INT	PK - NN - AI	
vendor_id	INT	NN - FK	
store_name	VARCHAR(100)	NN - UQ	
store_phone	VARCHAR(9)	NN	
store_email	VARCHAR(100)	NN - UQ	
description	MEDIUMTEXT	NN	
profile_picture	VARCHAR(250)		
street_address	VARCHAR(100)	NN	
city	VARCHAR(100)	NN	
postal_code	CHAR(8)	NN	
deleted	TINYINT	NN	0

- **store_review:** Avaliações das lojas.

Nome	Tipo	Atributos	Padrão / Expressão
store_review_id	INT	PK - NN - AI	
user_id	INT	NN - FK	
store_id	INT	NN - FK	
rating	INT	NN	
created	TIMESTAMP		CURRENT_TIMESTAMP
comment	MEDIUMTEXT	NN	

- **store_gallery:** Galeria de imagens das lojas.

Nome	Tipo	Atributos	Padrão / Expressão
store_gallery_id	INT	PK - NN - AI	
store_id	INT	NN - FK	
image_link	VARCHAR(250)	NN	

- **vendor:** Utilizadores que estão registados como vendedores.

Nome	Tipo	Atributos	Padrão / Expressão
vendor_id	INT	PK - NN - AI	
user_id	INT	NN - FK	
nif	CHAR(9)	NN - UQ	
deleted	TINYINT	NN	0

- **order:** Encomendas feitas pelos utilizadores.

Nome	Tipo	Atributos	Padrão / Expressão
order_id	INT	PK - NN - AI	
user_id	INT	NN - FK	
street_address	VARCHAR(100)	NN	
postal_code	CHAR(8)	NN	
city	VARCHAR(100)	NN	
created	TIMESTAMP		CURRENT_TIMESTAMP
comment	MEDIUMTEXT		
status	ENUM()	NN	

- **order_product:** Produtos que fazem parte das encomendas.

Nome	Tipo	Atributos	Padrão / Expressão
order_product_id	INT	PK - NN - AI	
order_id	INT	NN - FK	
product_id	INT	NN - FK	
price	DOUBLE	NN	
discount	DOUBLE	NN	0.0

3.1.2 | Diagrama EER

Introduzimos as tabelas criadas no *MySQL Workbench* e criámos os relacionamentos entre elas utilizando as chaves estrangeiras:

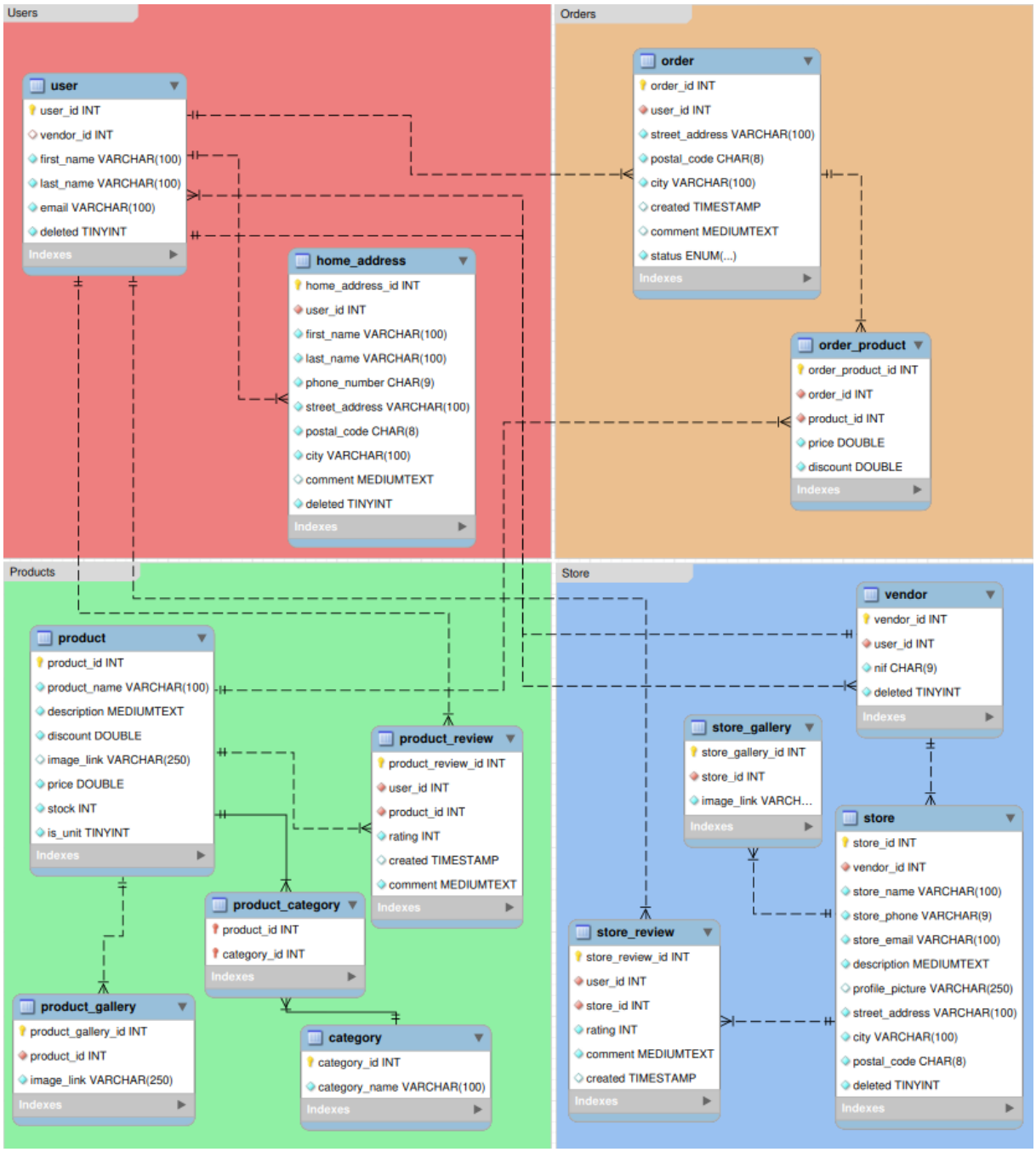

Figura 3.1: Diagrama EER

4 | Consultas á base de dados

Alguns exemplos de consultas que poderão ser feitas na aplicação:

- **Registo de Utilizador**

  O `user_id` é chave primária e será preenchido automaticamente de uma maneira sequencial. O `vendor_id` será inicializado sempre automaticamente como `NULL` pois quando um utilizador se regista inicialmente é apenas como um utilizador normal e não um vendedor. `deleted` também é inicializado automaticamente a `0`.

  Não será necessário armazenar qualquer palavra-passe ou token de autenticação, pois a estratégia de autenticação dos utilizadores será realizada através do *Auth0*, uma plataforma de gestão de autenticação.

```sql
-- Exemplo de registo de um utilizador
INSERT INTO user (first_name, last_name, email)
VALUES ('José', 'Manuel', 'jose.manuel@atec.pt');
```

- **Apagar utilizador**

  Apagar o utilizador com um certo `user_id`. Em vez de utilizarmos `DELETE FROM`, o campo `deleted` será definido como `1`, o que é uma prática mais segura que mantém a integridade dos dados e preserva o histórico.

```sql
-- Apagar o utilizador com id igual a 1
UPDATE user
SET deleted = 1
WHERE user_id = 1;
```

- **Consultar email de todos os vendedores**

  Mostrar o `email` de todos os utilizadores que são vendedores, ou seja, que tenham um `vendor_id` definido e que tenham o campo `deleted` definido como `0`, ou seja, que não tenham sido apagados.

```sql
-- Selecionar email de todos os utilizadores que são vendedores
SELECT email FROM user
WHERE deleted = 0 AND vendor_id IS NOT NULL;
```

TODO: fazer mais esta pagina com consultas

5 | Conclusão

TODO: Fazer conclusão