Diego Muller

Guilherme Massinhani

Luis Ricardo Holscher

Rubens Garcia Voivoda

Thiago Scaravonatto

Thiago Venturini

□ PixelMart □

Unoesc – SC

2024

Diego Muller

Guilherme Massinhani

Luis Ricardo Holscher

Rubens Garcia Voivoda

Thiago Scaravonatto

Thiago Venturini

□ PixelMart □

Trabalho realizado na disciplina de Banco de Dados I
Com intuito de aprendizagem e prática na manipulação
De dados e informações. Objeto de estudo do trabalho
Banco de dados para loja de jogos online.

Orientador: Prof. Roberson

Coordenador: Prof. Roberson

Unoesc – SC
2024

Sumário

RESUMO

O projeto PixelMart é um trabalho realizado em conjunto tendo apenas como intuito o aprendizado e a prática.

PixelMart tem em seu objetivo controlar eficientemente as vendas de jogos e administrar sessões, garantindo uma plataforma eficaz para operações relacionadas ao comércio e gestão de jogos online.

INTRODUÇÃO

PixelMart é um banco de dados projetado especificamente para atender às necessidades de gerenciamento de uma loja de jogos online. Ele é desenvolvido para armazenar e organizar dados relacionados a jogos, categorias, desenvolvedores, relacionamento com o cliente e vendas.

A estrutura do banco de dados PixelMart é composta por várias tabelas inter-relacionadas, que armazenam informações específicas sobre diferentes entidades do mundo dos jogos, além de ter toda a estrutura de uma empresa voltada para tal.

Através do uso desse banco de dados, a loja de jogos online PixelMart pode gerenciar informações como jogos disponíveis, categorias, desenvolvedores, vendas e muito mais. Isso permite que a loja ofereça um serviço eficiente, personalizado e fácil de usar para seus clientes, melhorando a experiência geral de compra e interação com a plataforma.

FUNÇÕES

Gestão de produtos: A principal função do PixelMart é gerenciar a venda de jogos, garantindo que o estoque disponível seja atualizado corretamente e não ultrapasse o limite.

Gestão de usuários: O PixelMart também gerencia os usuários cadastrados, permitindo que realizem compras. Além disso, o sistema assegura que as interações dos usuários sejam monitoradas para oferecer uma experiência personalizada e segura.

Histórico de compras: O sistema mantém um registro detalhado do histórico de compras dos usuários, permitindo que eles revisitem compras anteriores, façam download de jogos novamente e acompanhem suas atividades na loja.

Segurança de dados: O sistema garante a segurança dos dados dos usuários e transações, utilizando tecnologias avançadas de criptografia e seguindo as melhores práticas de segurança da informação.

REQUISITOS

Requisitos funcionais:

- Gerenciamento de vendas de jogos.
- Adicionar jogos novos facilmente.
- Permitir o cadastro, login e gerenciamento de contas de usuários.
- Permitir a adição, edição e remoção de jogos no inventário.

Requisitos não funcionais:

- O PixelMart deve ser compatível com diferentes navegadores web e dispositivos, garantindo uma experiência consistente em diversas plataformas.
- A plataforma deve ser responsiva e garantir tempos de resposta rápidos, mesmo durante períodos de pico de tráfego.
- As funcionalidades principais, como pesquisa de jogos, compra e avaliação, devem ser acessíveis com poucos cliques.

- Implementar um sistema de criptografia para proteger dados sensíveis dos usuários e transações financeiras.

COMANDOS DE CONSULTA

```
--1)
SELECT
   id_cliente,
   nome_cliente,
   endereco,
   telefone,
   email,
   cpf,
   data_nascimento
FROM
   cliente_final
WHERE
   EXTRACT(YEAR FROM AGE(data_nascimento)) BETWEEN 20 AND 30
   AND sexo = 'M'
ORDER BY
   nome_cliente DESC;


ALTER TABLE cliente_final ADD COLUMN data_nascimento DATE;

ALTER TABLE cliente_final ADD COLUMN sexo CHAR(1);

--2)
SELECT
   pedido.id_pedido,
   cliente_final.nome_cliente,
```

```
    cliente_final.cidade,
    pedido.data_pedido,
    pedido.status_pedido
FROM
    pedido
JOIN
    cliente_final ON pedido.id_cliente = cliente_final.id_cliente
WHERE
    EXTRACT(MONTH FROM pedido.data_pedido) IN (1, 3, 5, 7, 9, 11)
    AND EXTRACT(YEAR FROM pedido.data_pedido) = 2023
    AND cliente_final.cidade IN ('São Miguel do Oeste', 'Descanso')
ORDER BY
    pedido.data_pedido ASC;

ALTER TABLE cliente_final ADD COLUMN cidade VARCHAR(100);

--3)

SELECT
    lf.endereco AS cidade,
    lf.nome_loja,
    COUNT(lf.id_loja) OVER(PARTITION BY lf.endereco) AS total_lojas
FROM
    loja_fornecedora lf
JOIN
    catalogo_produto cp ON lf.id_loja = cp.id_loja
JOIN
    produto p ON cp.id_produto = p.id_produto
WHERE
    lf.endereco IN ('Maravilha', 'Descanso', 'Itapiranga', 'Guaraciaba')
```

```
    AND p.categoria = 'games para xbox'
ORDER BY
    total_lojas DESC, cidade, lf.nome_loja;

--4)

SELECT
    p.categoria AS tipo_game,
    lf.nome_loja,
    COUNT(p.id_produto) AS total_games
FROM
    loja_fornecedora lf
JOIN
    catalogo_produto cp ON lf.id_loja = cp.id_loja
JOIN
    produto p ON cp.id_produto = p.id_produto
WHERE
    p.categoria LIKE '%game%'
GROUP BY
    p.categoria, lf.nome_loja
ORDER BY
    total_games DESC, lf.nome_loja;
```

CRIAÇÃO DE TABELAS

```
-- Caso existam as determinadas tabelas antes, elas irão ser dropadas.
DROP TABLE IF EXISTS entrega CASCADE;
DROP TABLE IF EXISTS faturamento CASCADE;
DROP TABLE IF EXISTS pedido CASCADE;
```

```
DROP TABLE IF EXISTS estoque CASCADE;

DROP TABLE IF EXISTS catalogo_produto CASCADE;

DROP TABLE IF EXISTS cliente_final CASCADE;

DROP TABLE IF EXISTS produto CASCADE;

DROP TABLE IF EXISTS loja_fornecedora CASCADE;


-- Tabela De Lojas Fornecedoras.

CREATE TABLE loja_fornecedora (

id_loja SERIAL PRIMARY KEY,

nome_loja VARCHAR(80) NOT NULL,

endereco VARCHAR(80),

telefone VARCHAR(15),

email VARCHAR(50),

cnpj VARCHAR(15) UNIQUE

);


COMMENT ON COLUMN loja_fornecedora.id_loja IS 'Código da loja
fornecedora';

COMMENT ON COLUMN loja_fornecedora.nome_loja IS 'Nome da loja
fornecedora';

COMMENT ON COLUMN loja_fornecedora.endereco IS 'Endereço da loja
fornecedora';

COMMENT ON COLUMN loja_fornecedora.telefone IS 'Telefone de contato
da loja fornecedora';

COMMENT ON COLUMN loja_fornecedora.email IS 'Email da loja
fornecedora';

COMMENT ON COLUMN loja_fornecedora.cnpj IS 'CNPJ da loja
fornecedora';


-- Tabela De Produtos.
```

```
CREATE TABLE produto (
id_produto SERIAL PRIMARY KEY,
nome_produto VARCHAR(45) NOT NULL,
descricao VARCHAR(100),
categoria VARCHAR(30),
preco_base DECIMAL(10, 2),
fabricante VARCHAR(50)
);
COMMENT ON COLUMN produto.id_produto IS 'Código do produto';
COMMENT ON COLUMN produto.nome_produto IS 'Nome do produto';
COMMENT ON COLUMN produto.descricao IS 'Descrição do produto';
COMMENT ON COLUMN produto.categoria IS 'Categoria do produto';
COMMENT ON COLUMN produto.preco_base IS 'Preço base do produto';
COMMENT ON COLUMN produto.fabricante IS 'Fabricante do produto';

-- Tabela De Cliente Final.
CREATE TABLE cliente_final (
id_cliente SERIAL PRIMARY KEY,
nome_cliente VARCHAR(50) NOT NULL,
endereco VARCHAR(120),
telefone VARCHAR(15),
email VARCHAR(120),
cpf VARCHAR(20) UNIQUE
);

COMMENT ON COLUMN cliente_final.id_cliente IS 'Código do cliente final';
COMMENT ON COLUMN cliente_final.nome_cliente IS 'Nome do cliente
final';
```

```
COMMENT ON COLUMN cliente_final.endereco IS 'Endereço do cliente final';

COMMENT ON COLUMN cliente_final.telefone IS 'Telefone de contato do cliente final';

COMMENT ON COLUMN cliente_final.email IS 'Email do cliente final';

COMMENT ON COLUMN cliente_final.cpf IS 'CPF do cliente final';


-- Tabela De Pedido.

CREATE TABLE pedido (

id_pedido SERIAL PRIMARY KEY,

id_cliente INT NOT NULL,

data_pedido DATE,

status_pedido VARCHAR(50),

FOREIGN KEY (id_cliente) REFERENCES cliente_final(id_cliente)

);


COMMENT ON COLUMN pedido.id_pedido IS 'Código do pedido';

COMMENT ON COLUMN pedido.id_cliente IS 'Código do cliente final';

COMMENT ON COLUMN pedido.data_pedido IS 'Data em que o pedido foi realizado';

COMMENT ON COLUMN pedido.status_pedido IS 'Status atual do pedido';


-- Tabela de Faturamento.

CREATE TABLE faturamento (

id_faturamento SERIAL PRIMARY KEY,

id_pedido INT NOT NULL,

valor_total DECIMAL(10, 2),

data_faturamento DATE,

FOREIGN KEY (id_pedido) REFERENCES pedido(id_pedido)
```

```
);


COMMENT ON COLUMN faturamento.id_faturamento IS 'Código do faturamento';

COMMENT ON COLUMN faturamento.id_pedido IS 'Código do pedido';

COMMENT ON COLUMN faturamento.valor_total IS 'Valor total faturado no pedido';

COMMENT ON COLUMN faturamento.data_faturamento IS 'Data em que o faturamento foi realizado';


-- Tabela de Entrega.

CREATE TABLE entrega (

id_entrega SERIAL PRIMARY KEY,

id_pedido INT NOT NULL,

data_envio DATE,

data_entrega DATE,

status_entrega VARCHAR(50),

metodo_entrega VARCHAR(50),

FOREIGN KEY (id_pedido) REFERENCES pedido(id_pedido)

);


COMMENT ON COLUMN entrega.id_entrega IS 'Código da entrega';

COMMENT ON COLUMN entrega.id_pedido IS 'Código do pedido';

COMMENT ON COLUMN entrega.data_envio IS 'Data de envio da entrega';

COMMENT ON COLUMN entrega.data_entrega IS 'Data de entrega ao cliente final';

COMMENT ON COLUMN entrega.status_entrega IS 'Status atual da entrega';

COMMENT ON COLUMN entrega.metodo_entrega IS 'Método de entrega utilizado';
```

```
-- Tabela De Catálogo de Produtos.

CREATE TABLE catalogo_produto (

id_catalogo SERIAL PRIMARY KEY,

id_produto INT NOT NULL,

id_loja INT NOT NULL,

preco DECIMAL(10, 2),

data_inclusao DATE,

FOREIGN KEY (id_produto) REFERENCES produto(id_produto),

FOREIGN KEY (id_loja) REFERENCES loja_fornecedora(id_loja)

);


COMMENT ON COLUMN catalogo_produto.id_catalogo IS 'Código do
catálogo de produto';

COMMENT ON COLUMN catalogo_produto.id_produto IS 'Código do
produto';

COMMENT ON COLUMN catalogo_produto.id_loja IS 'Código da loja
fornecedora';

COMMENT ON COLUMN catalogo_produto.preco IS 'Preço do produto na loja
fornecedora';

COMMENT ON COLUMN catalogo_produto.data_inclusao IS 'Data de
inclusão do produto no catálogo';


-- Tabela De Estoque.

CREATE TABLE estoque (

id_estoque SERIAL PRIMARY KEY,

id_produto INT NOT NULL,

id_loja INT NOT NULL,

quantidade_disponivel INT,
```

FOREIGN KEY (id_produto) REFERENCES produto(id_produto),

FOREIGN KEY (id_loja) REFERENCES loja_fornecedora(id_loja)

);

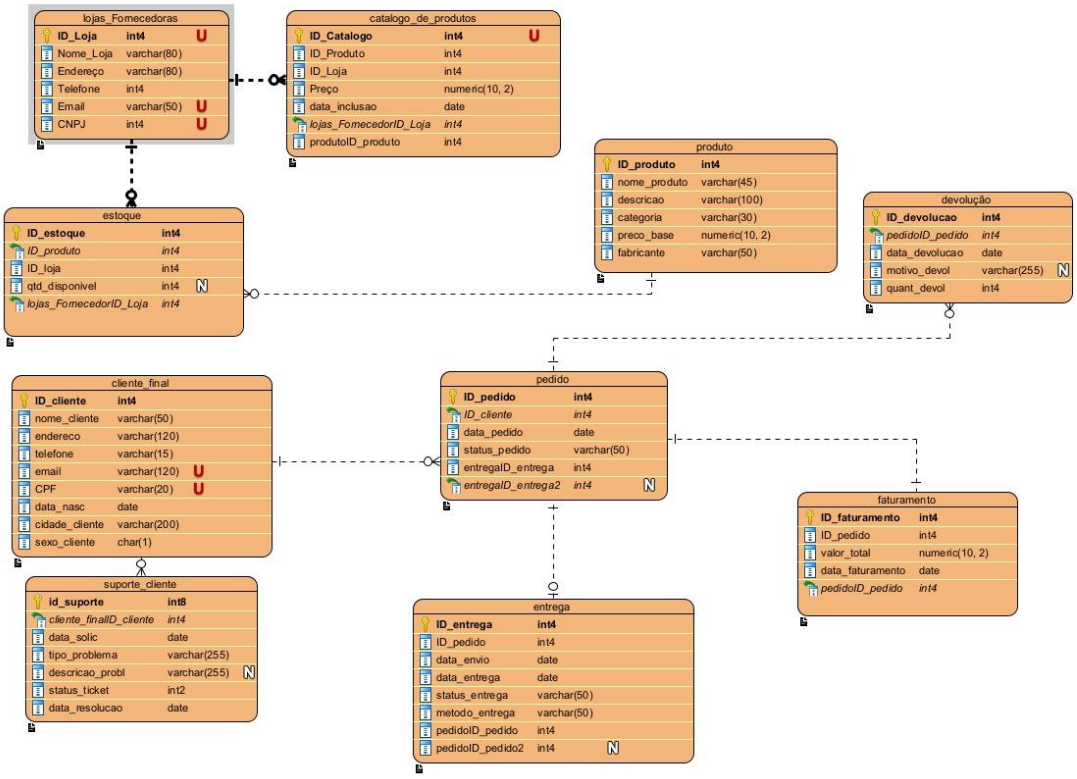
COMMENT ON COLUMN estoque.id_estoque IS 'Código do estoque';

COMMENT ON COLUMN estoque.id_produto IS 'Código do produto';

COMMENT ON COLUMN estoque.id_loja IS 'Código da loja fornecedora';

COMMENT ON COLUMN estoque.quantidade_disponivel IS 'Quantidade disponível do produto no estoque';

MODELO ENTIDADE RELACIONAL



CONCLUSÃO

O projeto PixelMart, teve como objetivo uma aplicação prática e eficiente para o gerenciamento de uma loja de jogos online, visto que estamos ao final do projeto, acreditamos que alcançamos esse objetivo com maestria. Conseguimos tal feito, através da criação de um banco de dados robusto e bem estruturado.

A estrutura do banco de dados, contribui para uma organização eficiente, ótimo funcionamento e fornece uma facilidade na manutenção do Bando de Dados. Esse sistema foi projetado para atender às necessidades específicas do comércio de jogos online, e nos auxiliou a nos desenvolvermos como equipe, além de contribuir imensamente para o aprendizado da matéria de Banco de Dados I.