

# TESTE TÉCNICO – EMPRESA WK TECHNOLOGY

CANDIDATO: HIGINO ALBEDIS DE ALMEIDA – albedis.almeida@gmail.com

## Objetivo:

Desenvolver uma cadastro de Pedido de Vendas, contendo seleção de Cliente e Produtos, com recursos para inclusão, alteração e exclusão (CRUD). Possibilitar a consulta de Pedidos anteriormente gravados e a exclusão completa do mesmo.

## Etapa 1 – Criação das tabelas

As tabelas foram criadas em banco de dados MYSQL/MARIADB, conforme a solicitação do teste. O arquivo DUMP se encontra na raiz da pasta do projeto com o nome DATABASE\_DUMP.SQL.

A versão utilizada do banco de dados foi 10.4.32-MariaDB, com arquivo para conexão “libmysql.dll” disponibilizado na pasta “library” do projeto.

The screenshot displays the phpMyAdmin interface for a database named 'wkdatabase'. The left sidebar shows the database structure with folders for 'cliente', 'pedido', 'pedido\_item', and 'produto'. The main panel shows the 'pedido\_item' table structure. The table has two columns: 'valor\_unitario' (decimal(10,2)) and 'valor\_total' (decimal(10,2)). The 'valor\_total' column is marked as 'STORED GENERATED'. Below the table structure, there is a section for 'Índices' (Indexes) showing three indexes: 'PRIMARY' (BTREE, Unique), 'pedidoitem\_pedido' (BTREE), and 'pedidoitem\_produto' (BTREE). The 'Partições' (Partitions) section shows 'Nenhum particionamento definido!'. The 'Informação' (Information) section provides details about the table's space usage and statistics.

Ação	Nome da chave	Tipo	Único	Pacote	Coluna	Cardinalidade	Colaço	Nulo	Comentário
<a href="#">Editar</a> <a href="#">Renomear</a> <a href="#">Eliminar</a>	PRIMARY	BTREE	Sim	Não	id_pedido_item	0	A	Não	
<a href="#">Editar</a> <a href="#">Renomear</a> <a href="#">Eliminar</a>	pedidoitem_pedido	BTREE	Não	Não	id_pedido	0	A	Não	
<a href="#">Editar</a> <a href="#">Renomear</a> <a href="#">Eliminar</a>	pedidoitem_produto	BTREE	Não	Não	id_produto	0	A	Não	

Uso do espaço		Estatísticas do registro	
Dados	16.0 KB	Formato	dinâmico
Índice	32.0 KB	Colaço	utf8_unicode_ci
Sobrecarga	0 Bytes	Próximo auto-índice	1
Efetivo	48.0 KB	Criação	28/08/2025 às 15:44
Total	48.0 KB	Última atualização	28/08/2025 às 15:48
<a href="#">Otimizar tabela</a>		Última verificação	28/08/2025 às 15:48

# TESTE TÉCNICO – EMPRESA WK TECHNOLOGY

CANDIDATO: HIGINO ALBEDIS DE ALMEIDA – albedis.almeida@gmail.com

## Etapa 2: Popular as tabelas

Foi utilizado o **COPILLOT** para gerar dados aleatórios para as tabelas de Clientes e Produtos. Esta foi a única colaboração da IA no projeto.

Hoje

considere o SQL de insert abaixo e crie 20 registros aleatórios. Apenas o campo "id\_cliente" é do tipo Integer, demais campos são respectivamente varchar(80), varchar(80) e char(2):

```
INSERT INTO `cliente`(`id_cliente`, `nome`, `cidade`, `uf`) VALUES ([value-1],[value-2],[value-3],[value-4])
```

Claro! Aqui estão 20 comandos **INSERT** com dados fictícios e variados, respeitando os tipos de dados que você especificou:

Sql

Copiar

```
INSERT INTO `cliente`(`id_cliente`, `nome`, `cidade`, `uf`) VALUES (1, 'Ana I', 'São Paulo', 'SP')
INSERT INTO `cliente`(`id_cliente`, `nome`, `cidade`, `uf`) VALUES (2, 'Carlo', 'Rio de Janeiro', 'RJ')
INSERT INTO `cliente`(`id_cliente`, `nome`, `cidade`, `uf`) VALUES (3, 'Fern', 'Belo Horizonte', 'MG')
INSERT INTO `cliente`(`id_cliente`, `nome`, `cidade`, `uf`) VALUES (4, 'João', 'Porto Alegre', 'RS')
INSERT INTO `cliente`(`id_cliente`, `nome`, `cidade`, `uf`) VALUES (5, 'Mari', 'Curitiba', 'PR')
INSERT INTO `cliente`(`id_cliente`, `nome`, `cidade`, `uf`) VALUES (6, 'Luca', 'Recife', 'PE')
INSERT INTO `cliente`(`id_cliente`, `nome`, `cidade`, `uf`) VALUES (7, 'Julia', 'Brasília', 'DF')
INSERT INTO `cliente`(`id_cliente`, `nome`, `cidade`, `uf`) VALUES (8, 'Rafa', 'Salvador', 'BA')
INSERT INTO `cliente`(`id_cliente`, `nome`, `cidade`, `uf`) VALUES (9, 'Cami', 'Fortaleza', 'CE')
INSERT INTO `cliente`(`id_cliente`, `nome`, `cidade`, `uf`) VALUES (10, 'Thia', 'Manaus', 'AM')
INSERT INTO `cliente`(`id_cliente`, `nome`, `cidade`, `uf`) VALUES (11, 'Pat', 'Goiânia', 'GO')
INSERT INTO `cliente`(`id_cliente`, `nome`, `cidade`, `uf`) VALUES (12, 'Bru', 'Natal', 'RN')
INSERT INTO `cliente`(`id_cliente`, `nome`, `cidade`, `uf`) VALUES (13, 'Lar', 'Vitória', 'ES')
INSERT INTO `cliente`(`id_cliente`, `nome`, `cidade`, `uf`) VALUES (14, 'Dieg', 'João Pessoa', 'PB')
INSERT INTO `cliente`(`id_cliente`, `nome`, `cidade`, `uf`) VALUES (15, 'Isal', 'Teresina', 'PI')
INSERT INTO `cliente`(`id_cliente`, `nome`, `cidade`, `uf`) VALUES (16, 'Gust', 'Aparecida de Goiânia', 'GO')
```

**TESTE TÉCNICO – EMPRESA WK TECHNOLOGY**  
CANDIDATO: HIGINO ALBEDIS DE ALMEIDA – albedis.almeida@gmail.com

Considere o SQL de insert abaixo e crie 20 registros aleatórios. Os campos são, respectivamente, Integer, varchar(80) e decimal(10,2).  
INSERT INTO `produto`(`id\_produto`, `descricao`, `preco\_venda`)  
VALUES ('[value-1]','[value-2]','[value-3]')

Claro, Albedis! Aqui estão 20 registros fictícios e variados para a tabela `produto`, com descrições comerciais e preços realistas formatados como `decimal(10,2)`:

Sql ^

 Copiar

```
INSERT INTO `produto`(`id_produto`, `descricao`, `preco_venda`) VALUES (1, 'Produto A', 123.45)
INSERT INTO `produto`(`id_produto`, `descricao`, `preco_venda`) VALUES (2, 'Produto B', 567.89)
INSERT INTO `produto`(`id_produto`, `descricao`, `preco_venda`) VALUES (3, 'Produto C', 987.65)
INSERT INTO `produto`(`id_produto`, `descricao`, `preco_venda`) VALUES (4, 'Produto D', 234.56)
INSERT INTO `produto`(`id_produto`, `descricao`, `preco_venda`) VALUES (5, 'Produto E', 789.01)
INSERT INTO `produto`(`id_produto`, `descricao`, `preco_venda`) VALUES (6, 'Produto F', 345.67)
INSERT INTO `produto`(`id_produto`, `descricao`, `preco_venda`) VALUES (7, 'Produto G', 678.90)
INSERT INTO `produto`(`id_produto`, `descricao`, `preco_venda`) VALUES (8, 'Produto H', 109.87)
INSERT INTO `produto`(`id_produto`, `descricao`, `preco_venda`) VALUES (9, 'Produto I', 456.78)
INSERT INTO `produto`(`id_produto`, `descricao`, `preco_venda`) VALUES (10, 'Produto J', 890.12)
INSERT INTO `produto`(`id_produto`, `descricao`, `preco_venda`) VALUES (11, 'Produto K', 210.34)
INSERT INTO `produto`(`id_produto`, `descricao`, `preco_venda`) VALUES (12, 'Produto L', 543.21)
INSERT INTO `produto`(`id_produto`, `descricao`, `preco_venda`) VALUES (13, 'Produto M', 987.65)
INSERT INTO `produto`(`id_produto`, `descricao`, `preco_venda`) VALUES (14, 'Produto N', 321.09)
INSERT INTO `produto`(`id_produto`, `descricao`, `preco_venda`) VALUES (15, 'Produto O', 654.32)
INSERT INTO `produto`(`id_produto`, `descricao`, `preco_venda`) VALUES (16, 'Produto P', 987.65)
INSERT INTO `produto`(`id_produto`, `descricao`, `preco_venda`) VALUES (17, 'Produto Q', 123.45)
```

# TESTE TÉCNICO – EMPRESA WK TECHNOLOGY

CANDIDATO: HIGINO ALBEDIS DE ALMEIDA – albedis.almeida@gmail.com

## Etapa 3 – Conexão com o banco de dados

Conforme solicitação no teste, um arquivo \*.INI foi disponibilizado para auxiliar na conexão com o banco de dados. Uma cópia do arquivo **dbconfig.ini** encontra-se na pasta library.

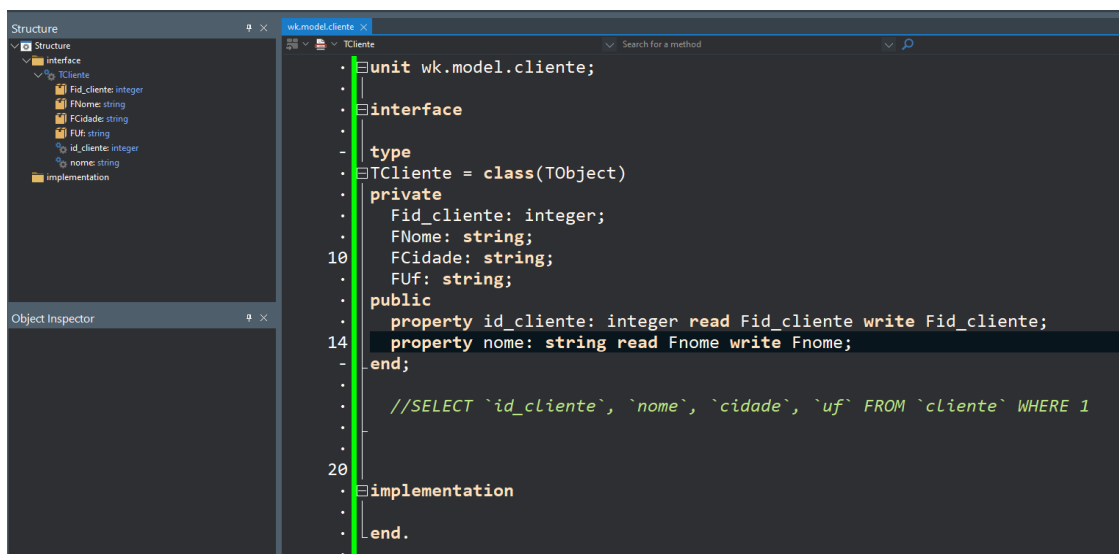
```
LIniPath := AppConfigIniPath;

if not FileExists(LIniPath) then
    raise Exception.CreateFmt('Arquivo INI não encontrado: %s', [LIniPath]);

LIni := TIniFile.Create(LIniPath);
try
    LDriverID      := LIni.ReadString('mysql', 'DriverID', 'MySQL');
    LVendorLib     := LIni.ReadString('mysql', 'VendorLib', 'libmysql.dll');
    LServer        := LIni.ReadString('mysql', 'Server', 'localhost');
    LPort          := LIni.ReadString('mysql', 'Port', '3306');
    LDatabase      := LIni.ReadString('mysql', 'Database', 'wkdatabase');
    LUser          := LIni.ReadString('mysql', 'UserName', 'root');
    LPass          := LIni.ReadString('mysql', 'Password', '');
    LCharset       := LIni.ReadString('mysql', 'CharacterSet', 'utf8mb4');
    LSSLMode       := LIni.ReadString('mysql', 'SSLMode', 'DISABLED');
```

## Etapa 4 – Criação de Modelos:

Foram criados modelos para as Entidades, facilitando assim a troca de informações entre os processos. Também crie classes de CONTROLLER, FACADE e SERVICE para separação das responsabilidades. Foi amplamente utilizado o princípio SOLID no que se refere a responsabilidade única.



**TESTE TÉCNICO – EMPRESA WK TECHNOLOGY**  
**CANDIDATO: HIGINO ALBEDIS DE ALMEIDA – albedis.almeida@gmail.com**

### Model Produto

```
unit wk.model.produto;
interface
type
TProduto = class (TObject)
private
    Fid_produto: integer;
    Fdescricao: string;
    Fpreco_venda: currency;
public
    property id_produto: integer
    property descricao: string
    property preco_venda: currency
end;

//`id_produto`, `descricao`, `preco_venda` FROM `produto` WHERE 1

implementation
end.
```

### Model Pedido



```
unit wk.model.pedido;
interface
type
TPedido = class(TObject)
private
    Fid_pedido: integer;
    Fdata_emissao: TDate;
    Fid_cliente: integer;
    Fvalor_total: currency;
public
    property id_pedido: integer read Fid_pedido write Fid_pedido;
    property data_emissao: TDate read fdata_emissao write fdata_emissao;
    property id_cliente: integer read fid_cliente write fid_cliente;
    property valor_total: currency read Fvalor_Total write fvalor_total;
end;
implementation
end.
```


**TESTE TÉCNICO – EMPRESA WK TECHNOLOGY**  
**CANDIDATO: HIGINO ALBEDIS DE ALMEIDA** – albedis.almeida@gmail.com


## Etapa 4: Criação das views

Abaixo temos na primeira figura a tela principal do projeto, em fase de desenvolvimento. Na sequência temos a tela principal concluída.

**Pedido de Venda**

Cliente    Pesquisar

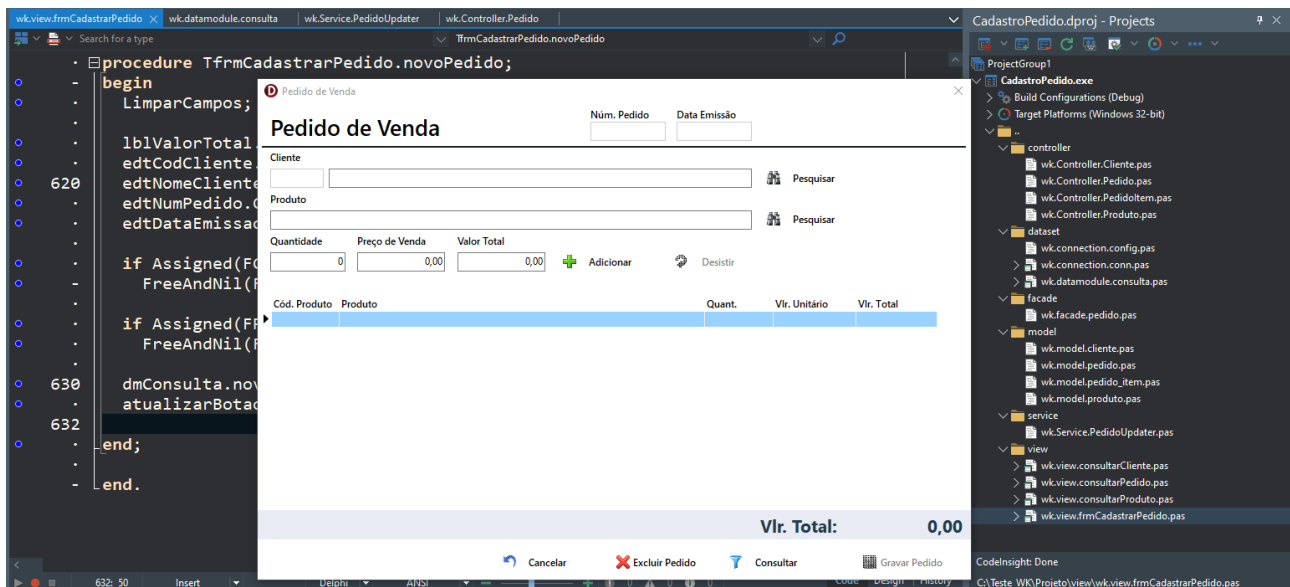
Produto   Pesquisar

Quantidade  Preço de Venda   Adicionar

# TESTE TÉCNICO – EMPRESA WK TECHNOLOGY

CANDIDATO: HIGINO ALBEDIS DE ALMEIDA – albedis.almeida@gmail.com

Tela Principal concluída



Todos os requisitos solicitados no teste foram minunciosamente verificados e atendidos. O projeto de TESTE TÉCNICO está completamente funcional e atende ao solicitado.

Certo de vossa atenção,

Higino Albedis de Almeida  
[albedis.almeida@gmail.com](mailto:albedis.almeida@gmail.com)

RJ – 30/08/2025