

NORMALIZAÇÃO

Normalização:

-Fizemos a normalização de uma tabela de cada vez, começando pela 1FN e avançando para a 2FN e 3FN.

-Eliminamos informações duplicadas para otimizar o armazenamento.

Resultados:

Tabelas Normalizadas:

Listamos as tabelas resultantes após a aplicação da normalização.

Explicitamos as chaves primárias e estrangeiras para facilitar a compreensão das relações entre as tabelas.

Conclusão

A normalização eficiente tornou o nosso projeto mais organizado, reduzindo redundâncias e melhorando a integridade do trabalho. Este processo é crucial para a manutenção a longo prazo do BD.

FORNECEDOR (id_fornecedor, nome, contato)

PRODUTOS (id_produto, estoque, nome, valor, id_fornecedor, id_categoria)

CATEGORIA (id_categoria, nome)

SUPERMERCADO (id_supermercado, nome, contato, endereco)

PEDIDOS (id_pedido, quantidade, id_supermercado)

PEDIDO (id_pedido, quantidade)

PEDIDO_PRODUTO (id_produto, id_pedido)

1a Forma Normal (1FN):

Para garantir a 1NF, certificamo-nos de que os atributos sejam atômicos:

FORNECEDOR: Nenhuma modificação necessária.

PRODUTOS: Nenhuma modificação necessária.

CATEGORIA: Nenhuma modificação necessária.

SUPERMERCADO: Nenhuma modificação necessária.

PEDIDOS: Nenhuma modificação necessária.

PEDIDO: Nenhuma modificação necessária.

PEDIDO_PRODUTO: Nenhuma modificação necessária.

NORMALIZAÇÃO

2a Forma Normal (2FN):

Nenhuma tabela possui uma chave primária composta, portanto, todas estão automaticamente na 2FN.

3a Forma Normal (3FN):

FORNECEDOR (id_fornecedor, nome, contato): Nenhuma dependência transitiva.

PRODUTOS (id_produto, estoque, nome, valor, id_fornecedor, id_categoria):
Nenhuma dependência transitiva.

CATEGORIA (id_categoria, nome): Nenhuma dependência transitiva.

SUPERMERCADO (id_supermercado, nome, contato, endereco): Nenhuma dependência transitiva.

PEDIDOS (id_pedido, quantidade, id_supermercado): Nenhuma dependência transitiva.

PEDIDO (id_pedido, quantidade): Nenhuma dependência transitiva.

PEDIDO_PRODUTO (id_produto, id_pedido): Nenhuma dependência transitiva.

Tabelas Normalizadas:

FORNECEDOR (id_fornecedor, nome, contato)

PRODUTOS (id_produto, estoque, nome, valor, id_fornecedor, id_categoria)

CATEGORIA (id_categoria, nome)

SUPERMERCADO (id_supermercado, nome, contato, endereco)

PEDIDOS (id_pedido, quantidade, id_supermercado)

PEDIDO (id_pedido, quantidade)

PEDIDO_PRODUTO (id_produto, id_pedido)

NORMALIZAÇÃO

FORNECEDOR

PK: id_fornecedor

PRODUTOS

PK: id_produto

FK: id_fornecedor (Referenciando FORNECEDOR)

FK: id_categoria (Referenciando CATEGORIA)

CATEGORIA

PK: id_categoria

SUPERMERCADO

PK: id_supermercado

PEDIDOS

PK: id_pedido

FK: id_supermercado (Referenciando SUPERMERCADO)

PEDIDO

PK: id_pedido

PEDIDO_PRODUTO

PK: id_produto, id_pedido

FK: id_produto (Referenciando PRODUTOS)

FK: id_pedido (Referenciando PEDIDO)

NORMALIZAÇÃO

```
1
2 CREATE TABLE cps.tbFornecedor (
3     id_fornecedor SERIAL PRIMARY KEY,
4     nome VARCHAR(255) NOT NULL,
5     contato VARCHAR(255) NOT NULL
6 );
7
8
9 CREATE TABLE cps.tbProdutos (
10     id_produto SERIAL PRIMARY KEY,
11     estoque INTEGER NOT NULL,
12     nome VARCHAR(255) NOT NULL,
13     valor DECIMAL(10, 2) NOT NULL,
14     id_fornecedor INTEGER REFERENCES cps.tbFornecedor(id_fornecedor),
15     id_categoria INTEGER REFERENCES cps.tbCategoria(id_categoria)
16 );
17
18
19 CREATE TABLE cps.tbCategoria (
20     id_categoria SERIAL PRIMARY KEY,
21     nome VARCHAR(255) NOT NULL
22 );
```

Query Query History

```
25 CREATE TABLE cps.tbSupermercado (
26     id_supermercado SERIAL PRIMARY KEY,
27     nome VARCHAR(255) NOT NULL,
28     contato VARCHAR(255) NOT NULL,
29     endereco VARCHAR(255) NOT NULL
30 );
31
32
33 CREATE TABLE cps.tbPedidos (
34     id_pedido SERIAL PRIMARY KEY,
35     quantidade INTEGER NOT NULL,
36     id_supermercado INTEGER REFERENCES cps.tbSupermercado(id_supermercado)
37 );
38
39 CREATE TABLE cps.tbPedido (
40     id_pedido SERIAL PRIMARY KEY,
41     quantidade INTEGER NOT NULL
42 );
43
44 CREATE TABLE cps.tbProdutosPedido (
45     id_produto INTEGER REFERENCES cps.tbProdutos(id_produto),
46     id_pedido INTEGER REFERENCES cps.tbPedido(id_pedido)
```

Data Output Messages Notifications

CREATE TABLE

Query returned successfully in 162 msec.

Total rows: 0 of 0

Query complete 00:00:00.162

Ln 25, Col 17

NORMALIZAÇÃO

The screenshot displays a database management tool interface. At the top, the connection is identified as 'pabd_vespertino/pabd@Flash abastecimentos'. Below the connection bar is a toolbar with icons for file operations, query execution, and other database functions. The main area is divided into two tabs: 'Query' and 'Query History'. The 'Query' tab is active, showing a SQL script with the following content:

```
27 nome VARCHAR(255) NOT NULL,  
28 contato VARCHAR(255) NOT NULL,  
29 endereco VARCHAR(255) NOT NULL  
30 );  
31  
32  
33 CREATE TABLE cps.tbPedidos (  
34     id_pedido SERIAL PRIMARY KEY,  
35     quantidade INTEGER NOT NULL,  
36     id_supermercado INTEGER REFERENCES cps.tbSupermercado(id_supermercado)  
37 );  
38  
39 CREATE TABLE cps.tbPedido (  
40     id_pedido SERIAL PRIMARY KEY,  
41     quantidade INTEGER NOT NULL  
42 );  
43  
44 CREATE TABLE cps.tbProdutosPedido (  
45     id_produto INTEGER REFERENCES cps.tbProdutos(id_produto),  
46     id_pedido INTEGER REFERENCES cps.tbPedido(id_pedido),  
47     PRIMARY KEY (id_produto, id_pedido)  
48 );
```

Below the query editor, there are three tabs: 'Data Output', 'Messages', and 'Notifications'. The 'Messages' tab is active, showing the message 'CREATE TABLE' and 'Query returned successfully in 162 msec.'.

At the bottom of the interface, there is a 'Tables (7)' section with a list of tables:

- > tbcategoria
- > tbfornecedor
- > tbpedido
- > tbpedidos
- > tbprodutos
- > tbprodutospedido
- > tbsupermercado