

Universidade Estadual do Rio Grande do Norte - UERN
Diretoria de Educação a Distância - DEAD
Sistemas para Internet
Banco de dados
Marcelo Henrique Lima Silva

PizzaLab

Sumário

1. Introdução
2. Estrutura do banco de dados
3. Procedimentos armazenados
4. Gatilhos
5. Scripts

1. Introdução

Neste documento contém o projeto do banco de dados para um sistema de pizzaria, que armazenará todas as informações nele contidas, desde de, usuários, produtos, estoque, etc. Tornando assim os dados armazenados para o sistema ainda mais eficientes para gestão e usabilidade, garantindo a integridade e disponibilidade dos dados.

O banco de dados possui relacionamento entre as tabelas criadas respeitando boas práticas de modelagem, além de outros recursos de armazenamento e gatilhos para automatizar as operações.

2. Estrutura do banco de dados

Dentro do sistema terá as seguintes tabelas:

- **Users:** Informações dos usuários, nome, telefone, endereço, tipo de usuário (Administrador, colaborador, cliente).
- **Products:** Informações dos tipos de pizzas do cardápio, nome, custo.
- **Ingredients:** Informações dos ingredientes contidos em cada pizza, nome, quantidade.
- **product_ingredients:** Relaciona os ingredientes com os produtos, incluindo informações de quantidade utilizada, unidade de medida.
- **stock:** Armazena as informações do estoque de ingredientes, incluindo quantidade disponível, unidade de medida, validade e data da última compra.
- **orders:** Informações de todos os pedidos, nome do produto, quantidade, tamanho, preço, valor da entrega, desconto.
- **items_order:** Informações de todos os itens dentro do pedido.

3. Procedimentos armazenados

Procedimentos armazenados serão utilizados para automatizar consultas.

- **PENDING_REQUEST():** Retorna todos os pedidos não concluídos ou cancelados.
- **PENDING_REQUEST_CLIENT(ID_CLIENT):** Retorna todos os detalhes do pedido do cliente.

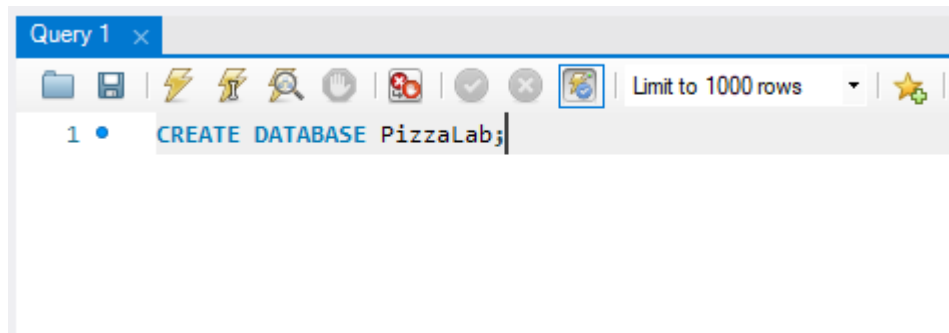
4. Gatilhos

Serão utilizados alguns gatilhos para automatizar algumas operações, incluindo garantir a integridade dos dados.

- **PENDING:** Define o status do pedido como pendente.
- **DELETE_NO:** Garante que nenhum pedido seja excluído do banco de dados.

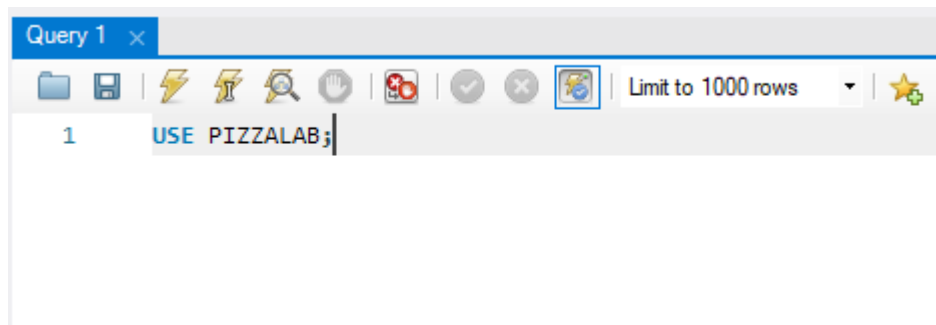
5. Scripts

Criando o banco de dados:



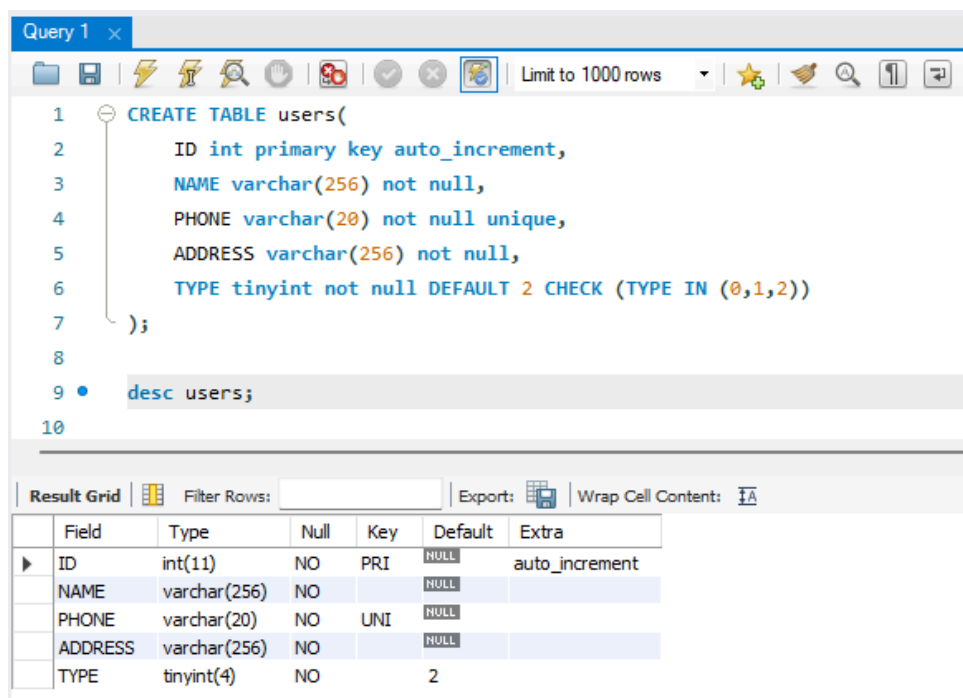
```
Query 1 x
1 • CREATE DATABASE PizzaLab;
```

Seleciona o banco de dados:



```
Query 1 x
1 USE PIZZALAB;
```

Criando a tabela **users**:



```
Query 1 x
1 CREATE TABLE users(
2     ID int primary key auto_increment,
3     NAME varchar(256) not null,
4     PHONE varchar(20) not null unique,
5     ADDRESS varchar(256) not null,
6     TYPE tinyint not null DEFAULT 2 CHECK (TYPE IN (0,1,2))
7 );
8
9 • desc users;
10
```

Result Grid

Field	Type	Null	Key	Default	Extra
ID	int(11)	NO	PRI	NULL	auto_increment
NAME	varchar(256)	NO		NULL	
PHONE	varchar(20)	NO	UNI	NULL	
ADDRESS	varchar(256)	NO		NULL	
TYPE	tinyint(4)	NO		2	

Criando tabela **products**:

```
Query 1 x
1 • CREATE TABLE products(
2     ID int primary key auto_increment,
3     NAME varchar(256) not null,
4     SIZE enum('P', 'M', 'G') not null,
5     PRICE decimal(10,2) not null
6 );
7
8 • desc products;
```

Field	Type	Null	Key	Default	Extra
ID	int(11)	NO	PRI	<div>NULL</div>	auto_increment
NAME	varchar(256)	NO		<div>NULL</div>	
SIZE	enum('P','M','G')	NO		<div>NULL</div>	
PRICE	decimal(10,2)	NO		<div>NULL</div>	

Criando a tabela **ingredients**:

```
Query 1 x
1 • CREATE TABLE ingredients(
2     ID int primary key auto_increment,
3     NAME varchar(100) not null unique
4 );
5
6 • desc ingredients;
```

Field	Type	Null	Key	Default	Extra
ID	int(11)	NO	PRI	<div>NULL</div>	auto_increment
NAME	varchar(100)	NO	UNI	<div>NULL</div>	

Criando a tabela **product_ingredients**:

Query 1 x

Limit to 1000 rows

```
1 • CREATE TABLE product_ingredients(  
2     ID int primary key auto_increment,  
3     ID_PRODUCT int not null,  
4     ID_INGREDIENT int not null,  
5     QUANTITY decimal(5,2) not null,  
6     UNIT varchar(20) not null,  
7     FOREIGN KEY (ID_PRODUCT) REFERENCES products(ID) ON DELETE CASCADE,  
8     FOREIGN KEY (ID_INGREDIENT) REFERENCES ingredients(ID) ON DELETE CASCADE  
9 );  
10  
11 • desc product_ingredients;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: [A](#)

Field	Type	Null	Key	Default	Extra
ID	int(11)	NO	PRI	NULL	auto_increment
ID_PRODUCT	int(11)	NO	MUL	NULL	
ID_INGREDIENT	int(11)	NO	MUL	NULL	
QUANTITY	decimal(5,2)	NO		NULL	
UNIT	varchar(20)	NO		NULL	

Criando a tabela **stock**:

Query 1 x

Limit to 1000 rows

```
1 • CREATE TABLE stock(  
2     ID int primary key auto_increment,  
3     ID_INGREDIENT int not null,  
4     QUANTITY decimal(10,2) not null CHECK (QUANTITY >= 0),  
5     UNIT varchar(20) not null,  
6     EXPIRATION_DATE DATE not null,  
7     LAST_PURCHASE TIMESTAMP DEFAULT CURRENT_TIMESTAMP,  
8     FOREIGN KEY (ID_INGREDIENT) REFERENCES ingredients(ID) ON DELETE CASCADE  
9 );  
10  
11 • desc stock;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: [A](#)

Field	Type	Null	Key	Default	Extra
ID	int(11)	NO	PRI	NULL	auto_increment
ID_INGREDIENT	int(11)	NO	MUL	NULL	
QUANTITY	decimal(10,2)	NO		NULL	
UNIT	varchar(20)	NO		NULL	
EXPIRATION_DATE	date	NO		NULL	
LAST_PURCHASE	timestamp	NO		current_timestamp()	

Criando a tabela **orders**:

Query 1

```
1 • CREATE TABLE orders(  
2     ID int primary key auto_increment,  
3     ID_USER int not null,  
4     DELIVERY_FEE decimal(10,2) not null DEFAULT 0.00,  
5     DISCOUNT decimal(10,2) not null DEFAULT 0.00,  
6     TOTAL decimal(10,2) not null DEFAULT 0.00,  
7     createdAt TIMESTAMP DEFAULT CURRENT_TIMESTAMP,  
8     FOREIGN KEY (ID_USER) REFERENCES users(ID) ON DELETE CASCADE  
9 );  
10  
11 • desc orders;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: |

Field	Type	Null	Key	Default	Extra
ID	int(11)	NO	PRI	NULL	auto_increment
ID_USER	int(11)	NO	MUL	NULL	
DELIVERY_FEE	decimal(10,2)	NO		0.00	
DISCOUNT	decimal(10,2)	NO		0.00	
TOTAL	decimal(10,2)	NO		0.00	
createdAt	timestamp	NO		current_timestamp()	

Criando a tabela **order_items**:

Query 1

```
1 • CREATE TABLE order_items(  
2     ID int primary key auto_increment,  
3     ID_ORDER int not null,  
4     ID_PRODUCT int not null,  
5     QUANTITY int not null CHECK (QUANTITY > 0),  
6     UNIT_PRICE decimal(10,2) not null,  
7     SUBTOTAL decimal(10,2) not null,  
8     FOREIGN KEY (ID_ORDER) REFERENCES orders(ID) ON DELETE CASCADE,  
9     FOREIGN KEY (ID_PRODUCT) REFERENCES products(ID) ON DELETE CASCADE  
10 );  
11  
12 • desc order_items;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: |

Field	Type	Null	Key	Default	Extra
ID	int(11)	NO	PRI	NULL	auto_increment
ID_ORDER	int(11)	NO	MUL	NULL	
ID_PRODUCT	int(11)	NO	MUL	NULL	
QUANTITY	int(11)	NO		NULL	
UNIT_PRICE	decimal(10,2)	NO		NULL	
SUBTOTAL	decimal(10,2)	NO		NULL	

Cadastrando usuários:

```
Query 1 x
Limit to 1000 rows
1 /* Criando um cliente padrão */
2 • INSERT INTO users (NAME, PHONE, ADDRESS) VALUES ('Carlos Souza', '999888777', 'Rua B, 456');
3 /* Criando um colaborador */
4 • INSERT INTO users (NAME, PHONE, ADDRESS, TYPE) VALUES ('Ana Pereira', '998877665', 'Rua C, 789', 1);
5 /* Criando um administrador */
6 • INSERT INTO users (NAME, PHONE, ADDRESS, TYPE) VALUES ('Dono da Pizzaria', '990011223', 'Rua D, 000', 0);
7
```

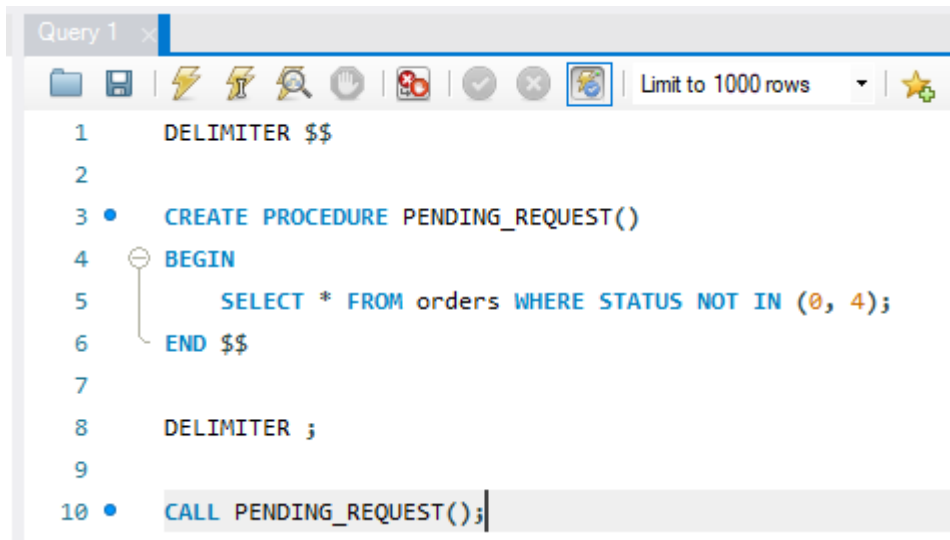
Cadastro e relacionamento do produto e ingrediente:

```
Query 1 x
Limit to 1000 rows
1 /* Cadastro de ingredientes */
2 • INSERT INTO ingredients (NAME) VALUES ('Queijo Mussarela'), ('Molho de Tomate'), ('Calabresa'), ('Orégano');
3 /* Cadastrando um produto */
4 • INSERT INTO products (NAME, SIZE, PRICE) VALUES ('Pizza de Calabresa', 'Média', 39.90);
5 /* Relacionando ingredientes com o produto */
6 • INSERT INTO product_ingredients (ID_PRODUCT, ID_INGREDIENT, QUANTITY, UNIT) VALUES
7 (1, 1, 150, 'g'), -- 150g de Queijo Mussarela
8 (1, 2, 100, 'ml'), -- 100ml de Molho de Tomate
9 (1, 3, 80, 'g'), -- 80g de Calabresa
10 (1, 4, 5, 'g'); -- 5g de Orégano
11
```

Criando pedido e adicionando itens:

```
Query 1 x
Limit to 1000 rows
1 /* Criando um pedido */
2 • INSERT INTO orders (ID_USER, DELIVERY_FEE, DISCOUNT, TOTAL)
3 VALUES (1, 5.00, 3.00, 76.80);
4 /* Adicionando itens ao pedido */
5 • INSERT INTO order_items (ID_ORDER, ID_PRODUCT, QUANTITY, UNIT_PRICE, SUBTOTAL)
6 VALUES (1, 1, 2, 39.90, 79.80);
7
```

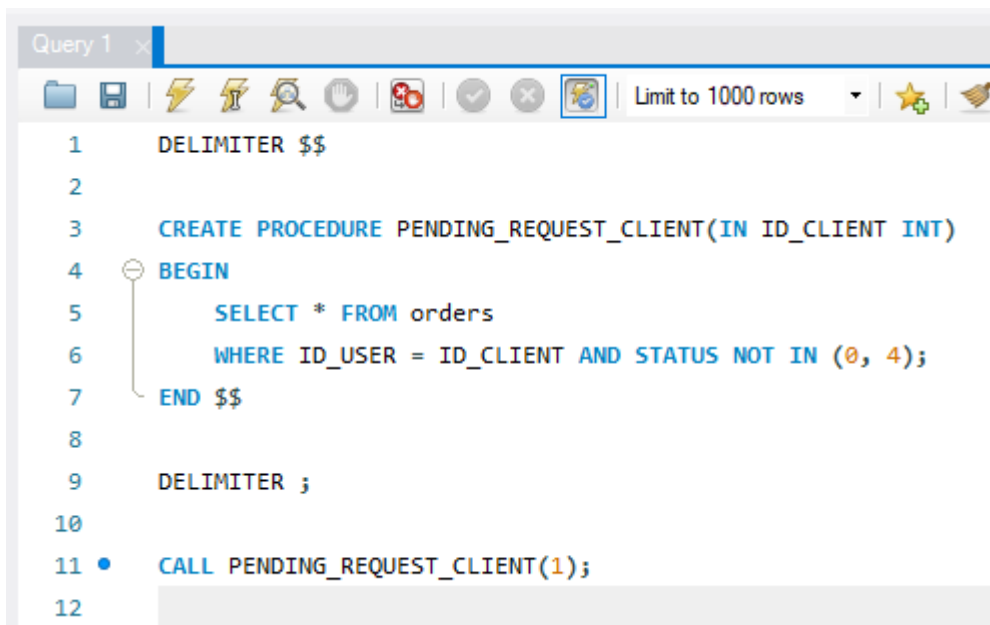

PENDING_REQUEST:



The screenshot shows a SQL query editor window titled "Query 1". The toolbar includes icons for file operations, execution, and a "Limit to 1000 rows" dropdown. The SQL code is as follows:

```
1 DELIMITER $$
2
3 • CREATE PROCEDURE PENDING_REQUEST()
4 BEGIN
5     SELECT * FROM orders WHERE STATUS NOT IN (0, 4);
6 END $$
7
8 DELIMITER ;
9
10 • CALL PENDING_REQUEST();
```

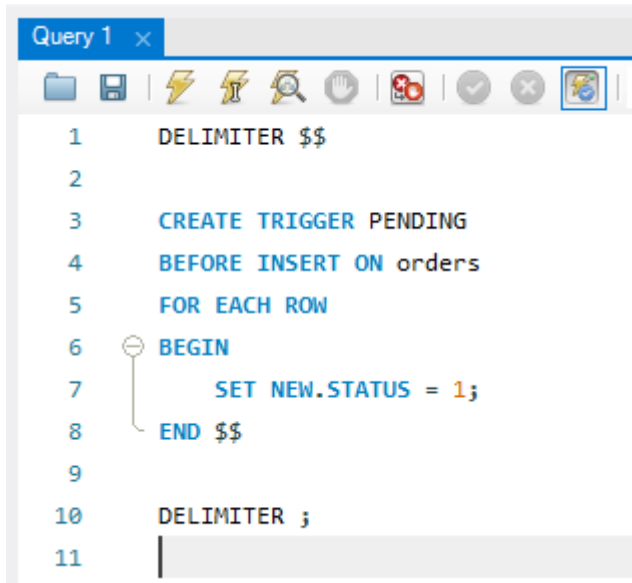
PENDING_REQUEST_CLIENT:



The screenshot shows a SQL query editor window titled "Query 1". The toolbar includes icons for file operations, execution, and a "Limit to 1000 rows" dropdown. The SQL code is as follows:

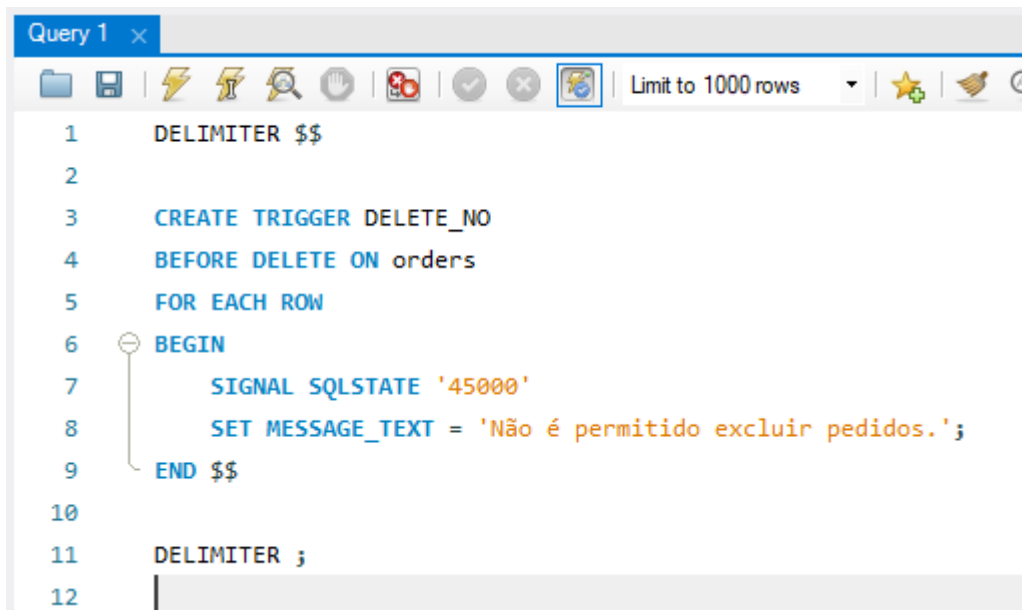
```
1 DELIMITER $$
2
3 CREATE PROCEDURE PENDING_REQUEST_CLIENT(IN ID_CLIENT INT)
4 BEGIN
5     SELECT * FROM orders
6     WHERE ID_USER = ID_CLIENT AND STATUS NOT IN (0, 4);
7 END $$
8
9 DELIMITER ;
10
11 • CALL PENDING_REQUEST_CLIENT(1);
12
```

PENDING:



```
Query 1 x
1 DELIMITER $$
2
3 CREATE TRIGGER PENDING
4 BEFORE INSERT ON orders
5 FOR EACH ROW
6 BEGIN
7     SET NEW.STATUS = 1;
8 END $$
9
10 DELIMITER ;
11
```

DELETE_NO:



```
Query 1 x
1 DELIMITER $$
2
3 CREATE TRIGGER DELETE_NO
4 BEFORE DELETE ON orders
5 FOR EACH ROW
6 BEGIN
7     SIGNAL SQLSTATE '45000'
8     SET MESSAGE_TEXT = 'Não é permitido excluir pedidos.';
9 END $$
10
11 DELIMITER ;
12
```