

Atividade do dia 27/08 - Sistema de Bancos de Dados

- Faça uma consulta filtrando um campo de texto utilizando like;

```
--Consulta com Like

select
*
from
produtos pr
where pr.nome_produto like '%2x2%'
```



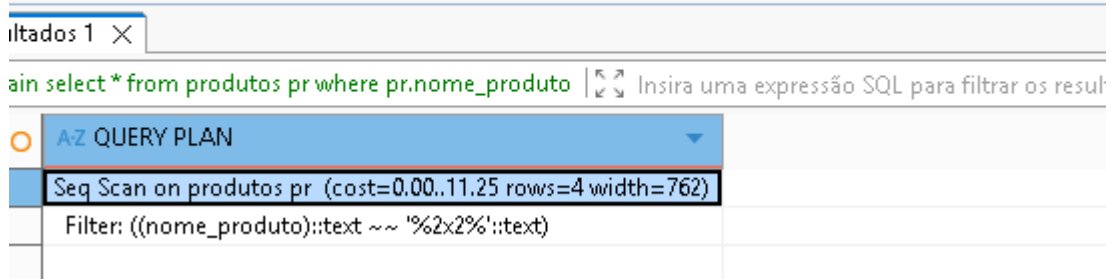
id_produto	nome_produto	categoria	preco	estoque	id_fornecedor
1	Cubo Mágico 2x2 Básico	2x2	15,9	100	1

Utilizei a consulta para verificar quais produtos dos que eu inseri possui a string “2x2” dentro dele, como um mecanismo de busca faria.

- Execute um comando explain e tire um print (anexe nesta atividade);

```
--Adicionando Explain

explain select * from produtos pr where pr.nome_produto like '%2x2%'
```



AZ QUERY PLAN
Seq Scan on produtos pr (cost=0.00..11.25 rows=4 width=762)
Filter: ((nome_produto)::text ~~ '%2x2%')::text

- Crie um index para a coluna que utilizou no filtro acima;

```
--Fazendo a consulta com index

create index idx_tagproduto
on produtos(nome_produto, preco)
```

Fiz esse index para uma ‘tag’ de produto com as informações que geralmente tem mais destaque de produtos em e-commerce.

- Refaça a primeira consulta e execute o explain novamente. Tire um novo print (anexe nesta atividade) e compare com o anterior. Aponte as diferenças.

Resultados 1 X	
explain select * from produtos pr where pr.nome_produto	
Grade	AZ QUERY PLAN
1	Seq Scan on produtos pr (cost=0.00..1.12 rows=1 width=762)
2	Filter: ((nome_produto)::text ~~ '%2x2%':text)

É possível reparar a diminuição do valor **cost** e **row** em comparação com a consulta antes de criar o índice.

- Altere uma coluna de varchar para int, avalie o retorno, inclusive se for erro;

```
CREATE TABLE fornecedores (
  id_fornecedor serial PRIMARY KEY,
  nome_empresa VARCHAR(100) NOT NULL,
  contato VARCHAR(100),
  email VARCHAR(100),
  pais VARCHAR(100),
  ativo BOOL
);

id_fornecedor serial PRIMARY KEY
nome_empresa VARCHAR(100) NOT NULL
contato int,
email VARCHAR(100),
pais VARCHAR(100),
ativo BOOLEAN
```

Resultou em um erro porque os inserts de contato são os nomes de contato em aspas simples:

```
INSERT INTO fornecedores (nome_empresa, contato, email) VALUES
('Shenzhen Cube Co', 'Li Wei', 'contact@szcube.cn'),
('Guangzhou Toys Ltd', 'Chen Yang', 'sales@gzt.cn'),
('Shanghai Speedcube', 'Wang Lei', 'info@ssc.cn'),
('Beijing Puzzle Import', 'Zhao Min', 'support@bpimport.cn'),
('Dongguan Cubes', 'Hu Xia', 'dg@cubeschina.cn'),
('Yiwu Trading', 'Chen Fang', 'yiwu@trade.cn'),
('Hong Kong Cubes', 'Lee Man', 'hk@cube.hk'),
('Ningbo Magic Co', 'Xu Ping', 'ningbo@magic.cn'),
('Suzhou Cube Export', 'Qin Yu', 'suzhou@cube.cn'),
('Hangzhou Toys', 'Sun Bo', 'hztoys@hz.cn');
```

Erro SQL [22P02]: ERRO: sintaxe de entrada é inválida para tipo integer:
"Li Wei"
Posição: 86

Posição do erro: line: 103 pos: 85

- Altere uma coluna de int para varchar avalie o retorno, inclusive se for erro;

```
CREATE TABLE itens_pedido (
  id_pedido INT,
  id_produto INT,
  quantidade varchar(255) not null,
  preco_unitario DECIMAL(10,2) NOT NULL,
  PRIMARY KEY (id_pedido, id_produto),
  FOREIGN KEY (id_pedido) REFERENCES pedidos(id_pedido),
  FOREIGN KEY (id_produto) REFERENCES produtos(id_produto)
);
```

```
INSERT INTO itens_pedido VALUES
```

```
(1, 2, 1, 35.50),
(1, 1, 2, 15.90),
(2, 3, 1, 55.00),
(3, 5, 1, 60.00),
(4, 6, 2, 40.00),
(5, 4, 1, 75.00),
(6, 7, 1, 45.00),
(7, 8, 1, 42.00),
(8, 9, 1, 48.00),
(9, 10, 1, 90.00);
```

Ao executar o insert, não notei nenhum erro significativo, provavelmente porque varchar suporta números, só não é possível talvez executar operações aritméticas e lógicas com eles.

- Crie um usuário com seu nome e dê todas as permissões de acesso para todas as tabelas;

```
--Criando usuário e dando acesso total
create user mathias with password 'bazinga'; --Cria o usuário
grant connect on database db_revenda_mathias to mathias; --Concede acesso ao banco
grant all privileges on all tables in schema public to mathias; --Concede todos os privilégios para todas as tabelas
ALTER DEFAULT PRIVILEGES IN SCHEMA public GRANT ALL ON TABLES TO mathias; --Concede todos os privilégios para todas as tabelas que serão criadas no futuro
```

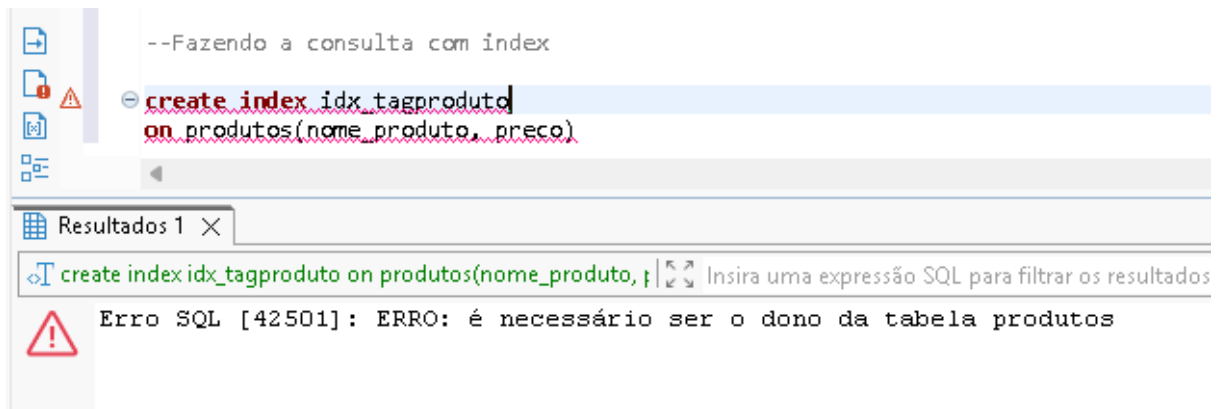
- Crie um usuário para seu colega apenas com permissão de select em uma das tabelas

A tabela escolhida foi a tabela **produto**

```
--Criando usuário colega e permissão de select
create user nicolas with password '23571113'
GRANT SELECT ON TABLE produtos TO nicolas;
```

- Refaça todos os items no usuário que criou para seu colega, registre tudo que ocorreu (erros e acertos)

Obtive sucesso na consulta com like e explain mas não obtive sucesso ao criar o index ou modificar os tipos de dados da tabela. Exemplo:



- De volta no seu usuário, crie 12 consultas, sendo 3 consultas semelhantes (somente com alteração do tipo de join: inner, left e right). Ou seja, são 4 consultas diferentes, sendo que cada consulta terá 3 versões, uma com cada tipo de join;

```

-- CONSULTAS COM INNERJOIN
SELECT pedidos.id_pedido, clientes.nome, produtos.nome_produto
FROM pedidos
INNER JOIN clientes ON pedidos.id_cliente = clientes.id_cliente
INNER JOIN itens_pedido ON pedidos.id_pedido = itens_pedido.id_pedido
INNER JOIN produtos ON itens_pedido.id_produto = produtos.id_produto;

--pedidos de cada cliente
SELECT clientes.nome, pedidos.id_pedido, pedidos.data_pedido
FROM clientes
INNER JOIN pedidos ON clientes.id_cliente = pedidos.id_cliente;

--produtos vendidos por fornecedor
SELECT fornecedores.nome_empresa, COUNT(produtos.id_produto) AS total_produtos
FROM fornecedores
INNER JOIN produtos ON fornecedores.id_fornecedor = produtos.id_fornecedor
GROUP BY fornecedores.nome_empresa;

--pagamentos e pedidos
SELECT pagamentos.id_pagamento, pedidos.id_pedido, pedidos.data_pedido
FROM pagamentos
INNER JOIN pedidos ON pagamentos.id_pedido = pedidos.id_pedido;

-- CONSULTAS COM RIGHTJOIN
SELECT pedidos.id_pedido, clientes.nome, produtos.nome_produto
FROM pedidos
RIGHT JOIN clientes ON pedidos.id_cliente = clientes.id_cliente
RIGHT JOIN itens_pedido ON pedidos.id_pedido = itens_pedido.id_pedido
RIGHT JOIN produtos ON itens_pedido.id_produto = produtos.id_produto;

SELECT fornecedores.nome_empresa, COUNT(produtos.id_produto) AS total_produtos
FROM fornecedores
RIGHT JOIN produtos ON fornecedores.id_fornecedor = produtos.id_fornecedor
GROUP BY fornecedores.nome_empresa;

SELECT pagamentos.id_pagamento, pedidos.id_pedido, pedidos.data_pedido
FROM pagamentos
RIGHT JOIN pedidos ON pagamentos.id_pedido = pedidos.id_pedido;

SELECT clientes.nome, pedidos.id_pedido, pedidos.data_pedido
FROM clientes
RIGHT JOIN pedidos ON clientes.id_cliente = pedidos.id_cliente;

-- CONSULTAS COM LEFTJOIN
SELECT pedidos.id_pedido, clientes.nome, produtos.nome_produto
FROM pedidos
LEFT JOIN clientes ON pedidos.id_cliente = clientes.id_cliente
LEFT JOIN itens_pedido ON pedidos.id_pedido = itens_pedido.id_pedido
LEFT JOIN produtos ON itens_pedido.id_produto = produtos.id_produto;

SELECT fornecedores.nome_empresa, COUNT(produtos.id_produto) AS total_produtos
FROM fornecedores
LEFT JOIN produtos ON fornecedores.id_fornecedor = produtos.id_fornecedor
GROUP BY fornecedores.nome_empresa;

SELECT pagamentos.id_pagamento, pedidos.id_pedido, pedidos.data_pedido
FROM pagamentos
LEFT JOIN pedidos ON pagamentos.id_pedido = pedidos.id_pedido;

SELECT clientes.nome, pedidos.id_pedido, pedidos.data_pedido
FROM clientes
LEFT JOIN pedidos ON clientes.id_cliente = pedidos.id_cliente;

```

Nota: Deu MUITO trabalho fazer isso (tive que pedir ajuda ao meu amigo gemini).

- **Atualize vários registros com colunas NULL;**

Não entendi muito bem esta instrução, não há nenhum insert null no meu banco de dados.

- Execute as consultas com Join novamente, avalie os resultados.

Bem, como não entendi muito bem como atualizar com colunas null, não há resultados para que eu possa avaliar, mas suponho que os dados devem ter sido atualizados e que seja possível observar isso na consulta.