



TECNOMAGOS

Dominando a magia dos Dados com SQL

Iara T. Bellodi

Livro para o Projeto DIO

Construção de material para estudo

Considerando a importância dos dados para qualquer empresa na atualidade, decidi criar um material de estudo sobre o uso de SQL com bancos de dados relacionais. A finalidade é proporcionar uma introdução aos comandos SQL para iniciantes.

Outra AI foi utilizada para gerar imagens uma vez que o uso do Midjourney não era mais gratuito na data de desenvolvimento deste esse projeto. A AI utilizada foi <https://gencraft.com/> para construção da imagem de capa usada neste projeto.

Dentro do projeto também fiz uso do ChatGPT para me auxiliar na construção de textos contidos nestes livro embora o material não tenha sido construído unicamente com esta ferramenta.

Os dados de tradução livre e exemplos de código foram escritos por mim considerando como referência livros acadêmicos da área de dados.

O título do livro foi a mistura de algumas das sugestões de título do ChatGPT considerando que eu adoro livros de ficção e magia

<https://showcode.app/> foi utilizado para mostrar o código.

Agradecimento

Agradeço ao meu marido Paulo H. Bellodi que me apoiou durante toda a minha jornada e que sempre acreditou em mim quando as vezes nem eu mesma acreditava.

Agradeço ao meu amigo Ricardo Abreu que sempre me indicou a área de tecnologia e que além de amigo fez o papel de terapeuta sempre disposto a me ouvir .

Meu muito obrigado ao Professor Felipe Silva Aguiar pela ideia do Projeto!
E claro, agradeço também ao Santander e a DIO por mais esta oportunidade de continuar aprendendo e me aprimorando, vocês foram incríveis e fazem a diferença!

01

**O SEU BANCO DE
DADOS
RELACIONAL**

Vamos começar pelo banco de dados

Neste livro trataremos da utilização dos comandos em um banco de dados relacional

Mas o que seria esse tal de banco relacional? Quando precisamos tratar de diversos dados, um banco de dados relacional é uma ótima pedida. Aqui daremos o exemplo do controle dos dados de uma loja de arte. A magia do banco de dados é te entregar as informações combinando-as de acordo com sua necessidade visando tratar um grande volume de dados com mais facilidade e eficiência.

Quando precisamos tomar uma decisão em uma empresa, os dados são primordiais, por isso você vê para todos os lados empresas com o termo **‘Data Driven’** que quer dizer que são orientadas a dados.

Criando seu banco de dados

Etapa de criação do nosso banco de dados deve ter como referência todos os dados de que precisaremos, para o exemplo de construção de nossos códigos em SQL teremos as tabelas cliente, pedido, produto, categoria, funcionarios, estoque, pedido_fornecedor, fornecedores.

Aqui como exemplo irei usar um Ateliê de arte e loja que vende diversos tipos de produto, quadros com pintura a óleo, pinturas aquarela, pintura em tecido e arte em biscuit. Essa loja não existe, é apenas um exemplo fantasioso para simularmos nosso banco de dados. Escolhi uma loja de arte porque é algo que eu amo, você pode criar outro exemplo para treinar.

Criando seu banco de dados

Nessa loja temos uma administradora Sabrina Santos e mais 4 artistas fantasticas, Leona Oliveira(responsável pelas pinturas), Amanda Meyer(biscuit), Felicia Santos(pintura em tecido) e Elisa Nascimento(bordados) . Bora criar a estrutura do nosso banco de dados?!

Para atender nossos clientes precisamos de lojas que forneçam os materiais para as artistas criarem, então dentre as tabelas que precisaremos criar temos a tabela fornecedores.

Bora pensar comigo, temos os funcionarios, os pedidos e os fornecedores. Os funcionarios produzem os itens pedidos através dos itens em estoque que são comprados dos fornecedores.

Criando seu banco de dados

Neste cenário eu imagino que você já possua instalado em sua máquina MySQL ou PostgreSQL. Caso contrario recomendo que procure no nosso maravilhoso YouTube as instruções para instalação de um deles que será de acordo com o seu Sistema operacional.

Recomendo o uso do PostgreSQL porque seu uso é gratuito.

02

COMANDOS SQL

Criando suas tabelas:

Geralmente o design da estrutura de banco de dados é feita por engenheiros mais experientes, para cargos mais iniciais, sua responsabilidade ficará mais voltada a manutenção e extração de dados.

Para extrair dados conforme a sua necessidade as tabelas compartilham relacionamentos através de chaves primarias e chaves estrangeiras, ou PRIMARY KEY e FOREIGN KEY. O tipo das Primary Key geralmente é int(numero inteiro) gerado automaticamente. Por exemplo, cada cliente terá sua primary key que é como o cpf que o identifica no sistema. Assim como o cliente os produtos vendidos na loja possuem uma primary key(chave primaria que os identifica), sendo assim, cada pedido dos clientes terá a chave primaria do pedido, e duas chaves estrangeiras(chave primaria do cliente e chave primaria do produto).

Criando a estrutura da tabela:

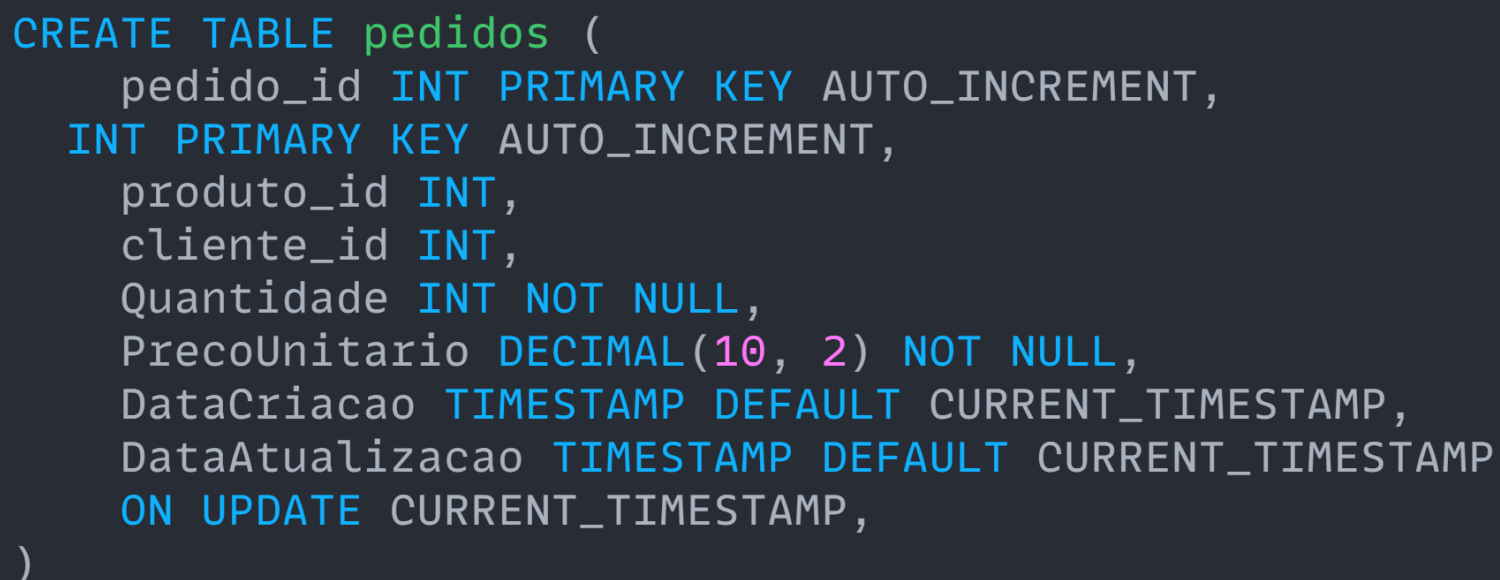
O exemplo abaixo mostra a estrutura para a criação da tabela clientes:

```
Untitled-1

CREATE TABLE clientes (
  Cliente_id INT PRIMARY KEY AUTO_INCREMENT,
  Nome VARCHAR(100) NOT NULL,
  CPF CHAR(11) NOT NULL UNIQUE,
  Pronome VARCHAR(20),
  Email VARCHAR(100),
  Telefone VARCHAR(15),
  Endereco VARCHAR(255)
  NUMERO VARCHAR(10),
  BAIRRO VARCHAR(100) NOT NULL,
  COMPLEMENTO(100) NOT NULL,
  CIDADE VARCHAR(50) NOT NULL,
  ESTADO VARCHAR(2),
  CEP INT
)
```

Criando a estrutura da tabela:

O exemplo abaixo mostra a estrutura para a criação da tabela pedidos:



```
CREATE TABLE pedidos (  
    pedido_id INT PRIMARY KEY AUTO_INCREMENT,  
    cliente_id INT PRIMARY KEY AUTO_INCREMENT,  
    produto_id INT,  
    cliente_id INT,  
    Quantidade INT NOT NULL,  
    PrecoUnitario DECIMAL(10, 2) NOT NULL,  
    DataCriacao TIMESTAMP DEFAULT CURRENT_TIMESTAMP,  
    DataAtualizacao TIMESTAMP DEFAULT CURRENT_TIMESTAMP  
    ON UPDATE CURRENT_TIMESTAMP,  
)
```

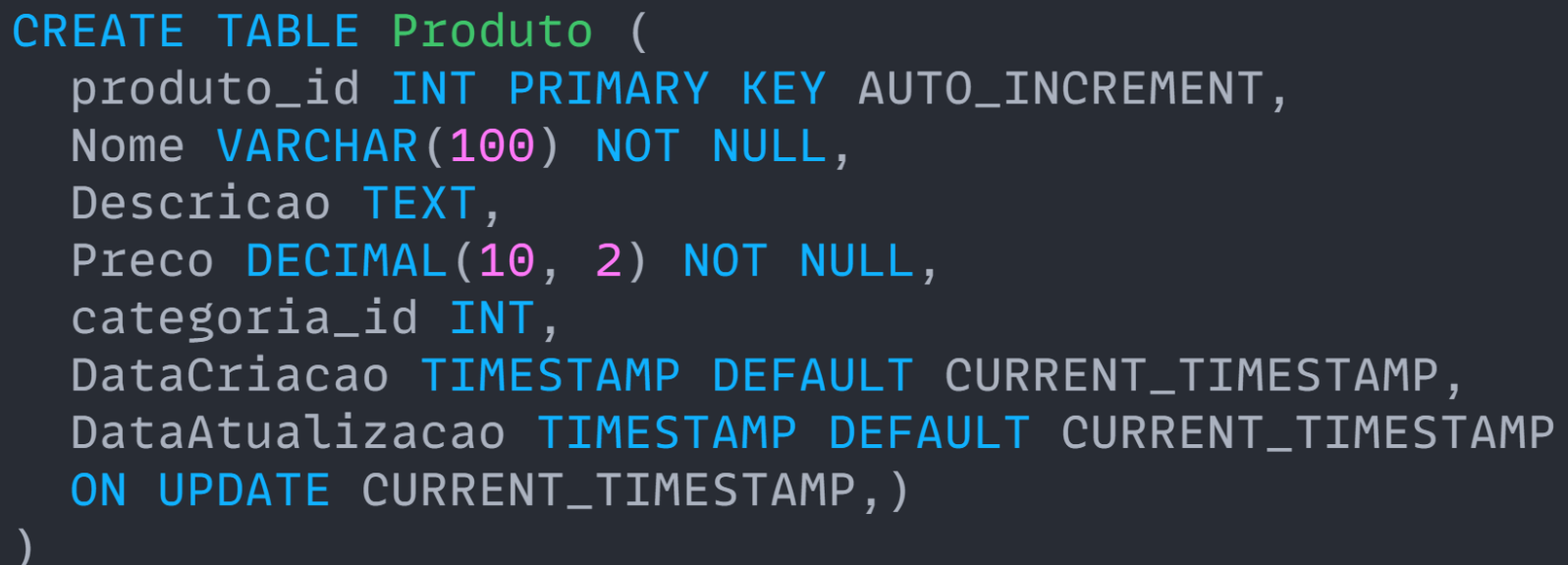
FOREIGN KEY (produto_id) referência

Produto(produto_id)

FOREIGN KEY (cliente_id) referência da tabela
clientes(cliente_id))

Criando a estrutura da tabela:

O exemplo abaixo mostra a estrutura para a criação da tabela cliente:



```
CREATE TABLE Produto (  
    produto_id INT PRIMARY KEY AUTO_INCREMENT,  
    Nome VARCHAR(100) NOT NULL,  
    Descricao TEXT,  
    Preço DECIMAL(10, 2) NOT NULL,  
    categoria_id INT,  
    DataCriacao TIMESTAMP DEFAULT CURRENT_TIMESTAMP,  
    DataAtualizacao TIMESTAMP DEFAULT CURRENT_TIMESTAMP  
    ON UPDATE CURRENT_TIMESTAMP,)  
)
```

FOREIGN KEY (categoria_id) referência da tabela categorias(categoria_id))

Tipos de dado:

Cada item descrito dentro do parenteses consiste em uma coluna com um tipo de item a ser inserido. Existem vários tipos de dado.

Como exemplo podemos citar na tabela Clientes temos a coluna '**Nome VARCHAR(100) NOT NULL**', dentro da coluna 'varchar' trata-se do tipo de conteúdo que será aceito dentro das células dessa coluna, neste caso varchar é um tipo variado de caracteres e o '**(100)**' consiste no número máximo de caracteres, '**not null**' significa que ela não poderá estar em branco.

Tipos de dado:

Quando queremos utilizar números existe mais de um tipo de dado, podendo ser **INT** que corresponde a números inteiros, **DECIMAL(10,2)** números decimais com 10 dígitos e até 2 dígitos depois da vírgula.

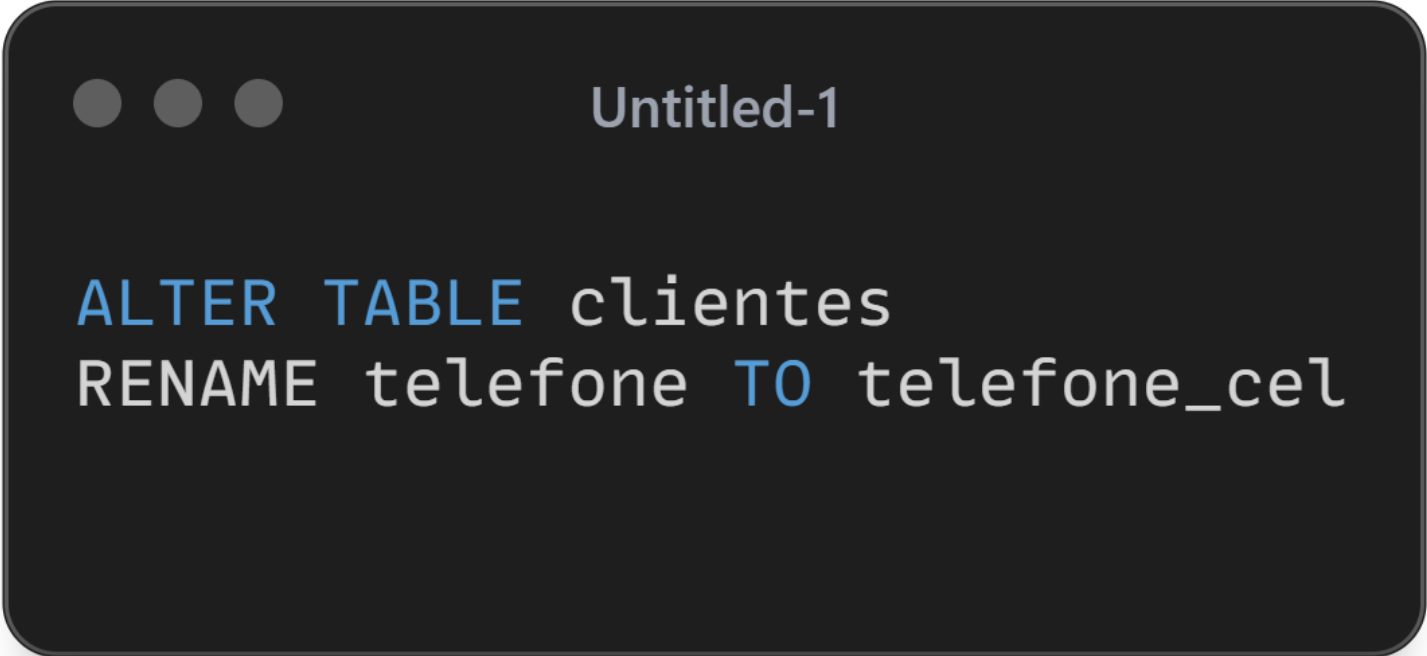
No caso de Telefone, utilizei VARCHAR por se tratar por considerar variados formatos de telephone com o simbolo de + e – e mesmo números mais longos, sendo que uma restrição muito especifica poderia corromper a integridade dos dados.

03

**MODIFICANDO OS
DADOS**

Alterando o nome de uma de uma coluna:

Ao construir a tabela Clientes me lembrei de que algumas pessoas tem mais de um telefone de contato e que seria melhor incluir um campo para telephone fixo e uma coluna para telephone celular, neste caso para alterar a tabela podemos usar o seguinte comando:



```
ALTER TABLE clientes  
RENAME telefone TO telefone_cel
```

Adicionando uma coluna:

Segue abaixo commando para adicionar uma coluna a tabela clientes

A dark-themed code editor window with a title bar containing three window control buttons (red, yellow, green) and the text 'Untitled-1'. The editor displays two lines of SQL code in a monospaced font with syntax highlighting: 'ALTER TABLE clientes' on the first line and 'ADD telefone_cel VARCHAR(20)' on the second line. The word 'ALTER' is blue, 'TABLE' is blue, 'clientes' is white, 'ADD' is blue, 'telefone_cel' is white, 'VARCHAR' is blue, and '(20)' is pink.

```
ALTER TABLE clientes  
ADD telefone_cel VARCHAR(20)
```

Inserindo dados em uma tabela:

Inserindo dados:

Para inserirmos dados em uma tabela devemos seguir o comando INSERT INTO nome_da_tabela(coluna,coluna_2...) VALUES(item1,item2,item3...) ,seguindo a ordem colunas para inserir os tabela dos itens que devem ser adicionados:

```
INSERT INTO produto (nome,descricao,preco,categoria)
VALUES ('raposa','raposa em meio a flores',299.99, 'pintura a oleo')
```

SELECIONANDO DADOS:

Sabrina recebeu o contato de uma cliente que pretende sair do país e pediu para ter seus dados removidos de nosso banco de dados: Para fazer isso Sabrina criou o seguinte código.

Sempre que for usar este comando verifique a condição após WHERE para que somente as linhas que atendam essa condição na coluna informada sejam apagadas.



```
DELETE FROM clientes  
WHERE nome = 'Eloisa Ortega'
```


SOBRE DROP:


O comando **DROP** no SQL é utilizado para remover completamente objetos do banco de dados, como tabelas, bancos de dados, índices, colunas, procedimentos armazenados, e outros objetos. Quando um objeto é removido usando o DROP, ele é permanentemente deletado e não pode ser recuperado a menos que exista um backup.

04

**SELECCIONANDO
DADOS**

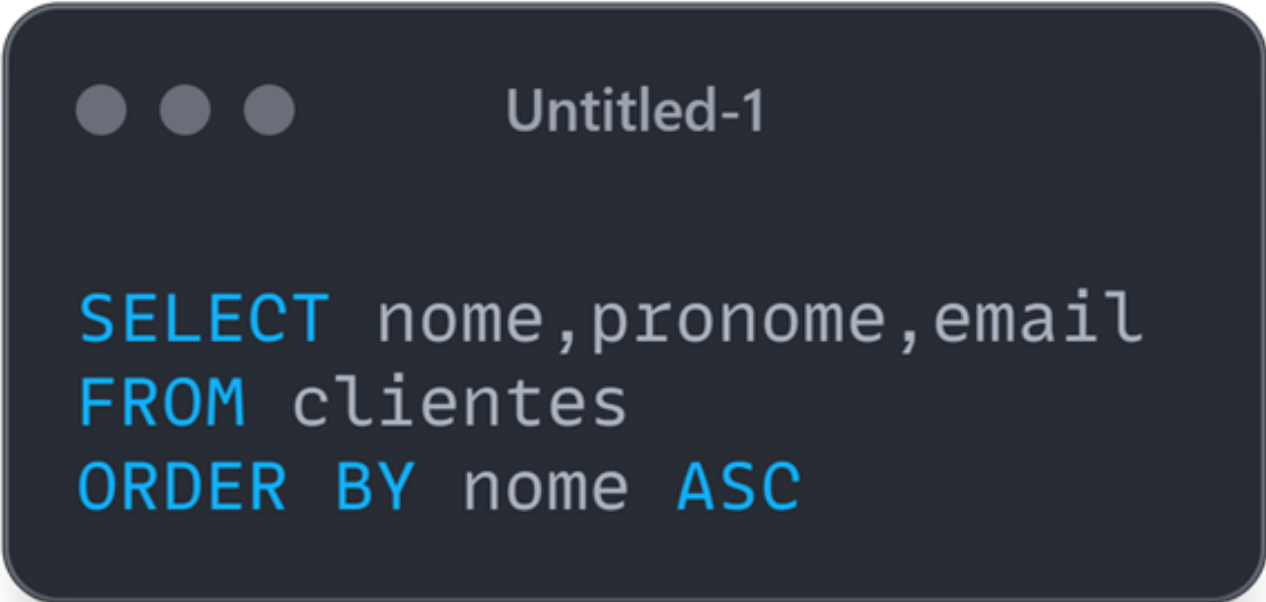
SELECIONANDO DADOS:

Comando para quando desejamos ver todos os conteúdos de uma tabela:



```
SELECT * FROM clientes
```

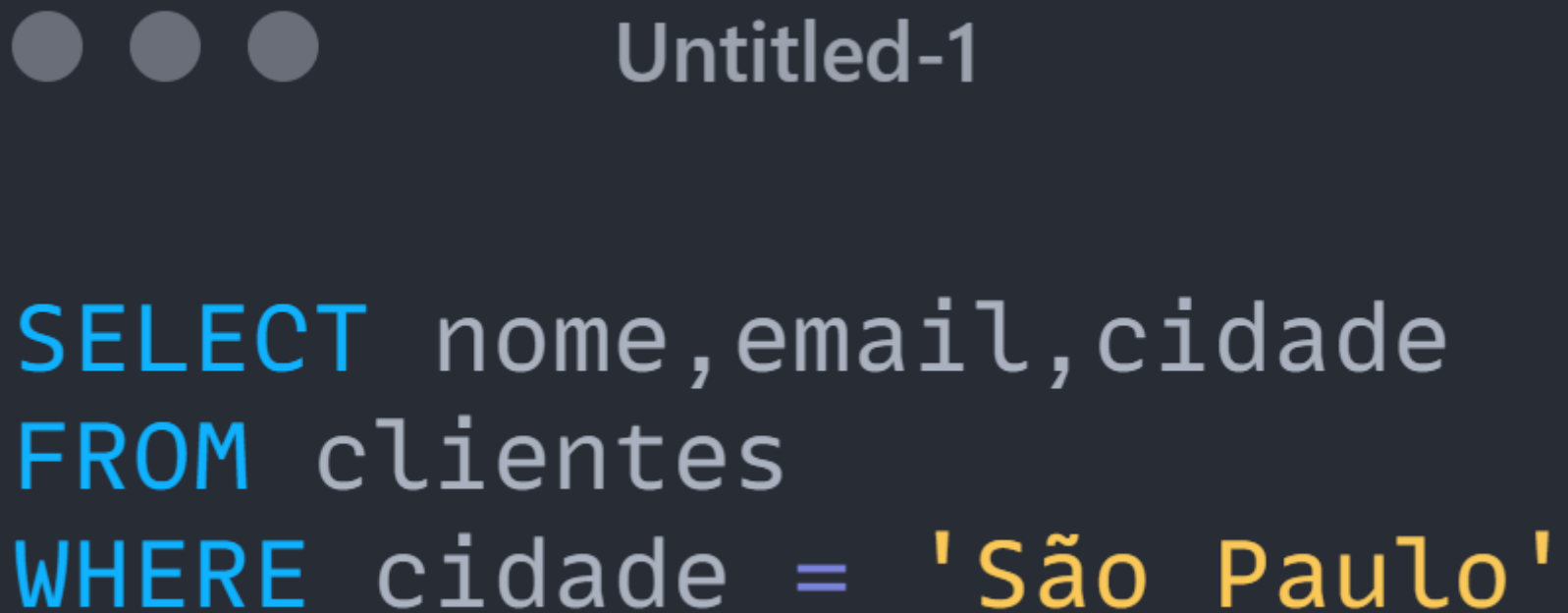
Para aumentar as vendas decidimos mandar e-mails para nossos clientes oferecendo descontos, para apenas visualizar os nomes dos clientes, seus pronomes e seus emails, considerando a coluna nomes em ordem alfabética devemos usar o seguinte comando:



```
SELECT nome, pronome, email  
FROM clientes  
ORDER BY nome ASC
```

SELECIONANDO DADOS:

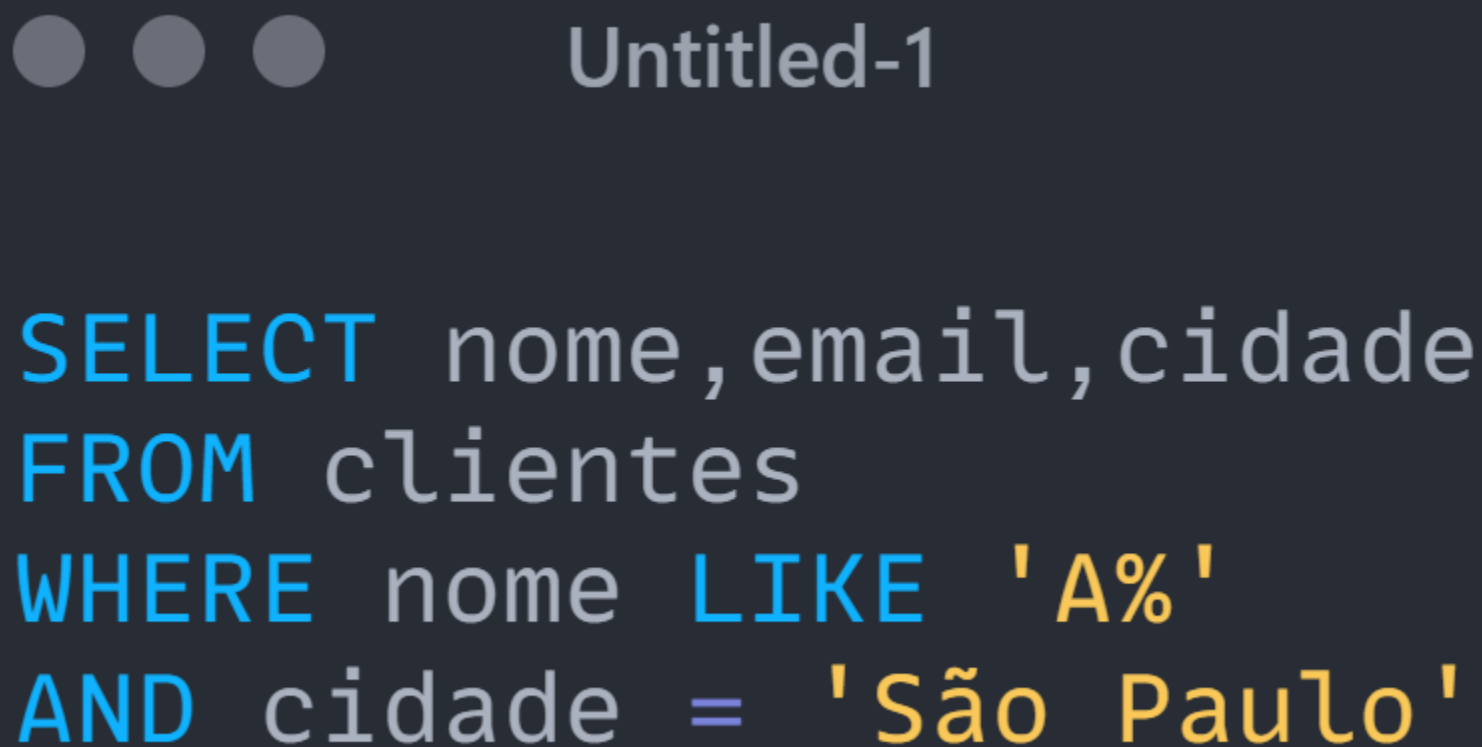
A empresa fez uma parceria com um outro atelie de arte, sendo assim decidimos convidar nossos clients para um evento na mesma cidade do atelie para uma exposição, Para saber se temos clients na mesma cidade do atelie podemos fazer o seguinte comando



```
SELECT nome,email,cidade  
FROM clientes  
WHERE cidade = 'São Paulo'
```

SELECIONANDO DADOS:

Quando Sabrina foi ao banheiro Leona atendeu o telefone da loja para ajudar e anotou um recado, porem se esqueceu completamente quem era a cliente, ela só se lembra que o nome da cliente se começa com A e que ela falou que estaria na cidade de volta a sua casa em São Paulo. Para tentar refrescar a memória de Leona, Sabrina criou o seguinte código



```
SELECT nome,email,cidade
FROM clientes
WHERE nome LIKE 'A%'
AND cidade = 'São Paulo'
```

SELECIONANDO DADOS:

O LIKE serve para pegar trechos do conteúdo de uma célula , usando LIKE 'A%' eu disse para o sistema que eu quero que me mostre os itens desta coluna que começam com a letra A não importando o que vem a seguir.

Se eu quisesse procurar pelo final do nome, por exemplo alguém com o final do nome 'bete' eu poderia usar LIKE '%bete'.

A função do AND é para restringir a pesquisa a clients com o nome que começam com a letra 'A' e moram na cidade de São Paulo.

Caso eu quisesse que fosse alguém cujo nome começa com a letra 'A' ou qualquer pessoa que morasse em 'São Paulo' , no lugar do AND eu utilizaria a opção OR.

Clientes VIPS:

Decidimos reunir alguns dos nossos clientes em um evento, para isso podemos selecionar os 10 clientes que mais compram com a loja. Para isso utilizamos o JOIN unindo dados de duas tabelas:

```
SELECT
    c.cliente_id,
    c.nome,
    c.email,
    COUNT(p.pedido_id) AS numero_de_pedidos
FROM
    clientes c
INNER JOIN
    pedidos p ON c.cliente_id = p.cliente_id
GROUP BY
    c.cliente_id,
    c.nome,
    c.email
ORDER BY
    numero_de_pedidos DESC
LIMIT 10;
```

Dica:

As vezes compesa utilizar uma CTE para facilitar a organização de dados e extrair a informação de que você precisa. Uma CTE nada mais é do que uma planilha temporaria para uso de dados especificos.

Gpsto de utilizar essa funcionalidade quando quero combiner dados de tuas tabelas, mas não quero utilizar o conteúdo de toda a tabela em questão.

```
Untitled-1

WITH CategoriaPedidos AS (
    SELECT
        c.categoria_id,
        c.descricao_produto,
        SUM(p.quantidade) AS total_quantidade
    FROM
        pedido p
    JOIN
        produto pr ON p.produto_id = pr.produto_id
    JOIN
        categoria c ON pr.categoria_id = c.categoria_id
    GROUP BY
        c.categoria_id,
        c.descricao
)

-- Selecionando a categoria mais pedida
SELECT
    categoria_id,
    descricao_produto,
    total_quantidade
FROM
    CategoriaPedidos
ORDER BY
    total_quantidade DESC
LIMIT 1;
```

Dica:

Talvez o Código tenha ficado um pouco confuso com o uso de JOIN, mas para facilitar nós colocamos apelidos nas tabelas, e para dizer ao Sistema de onde vinham cada coluna inserimos a inicial correspondente a tabela escolhida seguida de um ponto ao nome das colunas.

Veja o trecho de Código abaixo para entender melhor:

```
WITH CategoriaPedidos AS (  
    SELECT  
        c.categoria_id,  
        c.descricao_produto,  
        SUM(p.quantidade) AS total_quantidade  
    FROM  
        pedido p  
    JOIN  
        produto pr ON p.produto_id = pr.produto_id
```

Aqui a tabela pedido é apelidada de 'p' quando colocamos a letra na sequencia logo após o nome da tabela.

Dica:

Recomendo o uso de uma CTE sempre que precisar combinar dados temporariamente:

```
Untitled-1

WITH ProdutoCategoria AS (
    SELECT
        p.produto_id,
        p.categoria_id,
        p.valor,
        p.nome as produto,
        c.nome AS nome_categoria
    FROM
        produto p
    JOIN
        categoria c ON p.categoria_id = c.categoria_id
)

-- Selecionando os dados da CTE
SELECT
    produto,
    nome_categoria,
    valor
FROM
    ProdutoCategoria
```

Existem outros tipos de JOIN que podem ser usados e recomendo que pesquise as diversas possibilidades de combinação de tabelas.

Dica:

Abaixo eu selecionei os produtos com valores superiores a 200 reais, sendo que o WHERE nos permite essa comparação:

```
Untitled-1

WITH ProdutoCategoria AS (
    SELECT
        p.produto_id,
        p.categoria_id,
        p.valor,
        p.nome AS produto,
        c.nome AS nome_categoria
    FROM
        produto p
    JOIN
        categoria c ON p.categoria_id = c.categoria_id
)

-- Selecionando os dados da CTE
SELECT
    produto,
    nome_categoria,
    valor
FROM
    ProdutoCategoria
WHERE
    valor > 200.00
```

05

**APRENDER É UM
PROCESSO
CONTINUO**

Dicas de sites para aprender mais comandos SQL e banco de dados:

Bora continuar com os estudos!

Canais que oferecem um ótimo material para começar os estudos:

Plataforma da DIO

<https://www.dio.me/>

YouTube:

Playlist do Gustavo Guanabara – MySQL

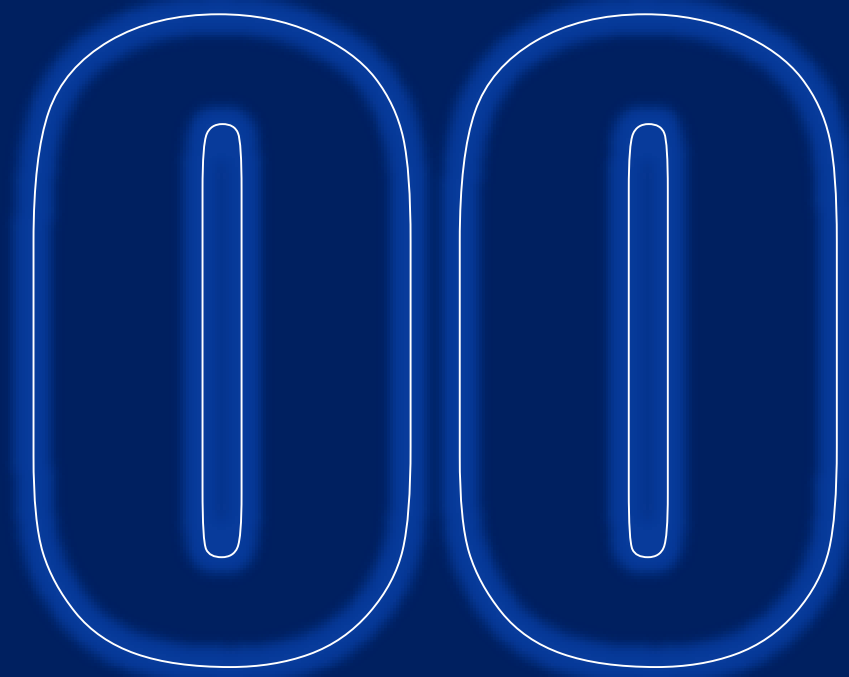
https://www.youtube.com/playlist?list=PLHz_AreHm4dkBs-795Dsgvau_ekxg8g1r

Para quem já sabe inglês indico

Freecodecamp

https://www.youtube.com/playlist?list=PLHz_AreHm4dkBs-795Dsgvau_ekxg8g1r





TRADUÇÃO LIVRE DOS COMANDOS

***Para quem começou a
aprender SQL mas não sabe
muito inglês**

Tradução Livre para facilitar a compreensão do seu Código:

CREATE TABLE → Crie tabela(você usa esse comando para criar uma tabela com as especificações que você deseja)

DROP TABLE → **DELETE ESSA TABELA**(para quando você quer deletar uma tabela por completo)

DELETE FROM [a_tabela] WHERE → **DELETE ONDE**(Para deletar as linhas na tabela considerando uma condição específica. Lembre do WHERE por favor, isso pode fazer toda a diferença)

Tradução Livre para facilitar a compreensão do seu Código:

SELECT → Selecione (para você selecionar as colunas que você deseja)

FROM → DE(indique a tabela a ser consultada)

WHERE → ONDE(aqui você cria a condição, você deseja ver os valores que atendem a alguma condição, por exemplo coluna que tenha valores iguais a alguma coisa)

LIKE → ASSIM COMO (valor similar ou que contenha determinado aspecto)

AND → (Condição a ser atendida)

OR → OU(Pode ser um valor ou outro)

Tradução Livre para facilitar a compreensão do seu Código:

ALTER TABLE → ALTERE A TABELA (para alterar uma tabela já existente)

ADD COLUMN → ADICIONE COLUNA (para ser usado em conjunto com ALTER TABLE)

VARCHAR → CARACTERES (para criar uma coluna com caracteres mais diversos)

JOIN ON → JUNTE EM (para unir conteúdo de duas tabelas utilizando suas chaves primarias e estrangeiras)

Traduções Livre para facilitar a compreensão do seu Código:

MAX → MAXIMO(para obter o valor máximo de uma coluna)

MIN → MENOR(para obter o menor valor de uma coluna)

GROUP BY → JUNTE POR(para juntar um tipo de dado)

ORDER BY → ORDENE POR(para ordenar por uma determinada coluna em questão)

DESC → DECRESCENTE(Ordenar do maior para o menor para usar em conjunto com order by)

ASC → (Classifica do valor mais baixo para o mais alto)

Tradução Livre para facilitar a compreensão do seu Código:

WITH[nome_da_tabela_temporaria] **AS** →

COM COMO(Consiste em uma CTE ou seja uma tabela temporaria criada unicamente para facilitar a visualização de dados)

CASE WHEN THEN ELSE→**QUANDO**

ASSIM(Condição a ser analisada) ENTÃO(então você diz o que espera que ele faça) CASO CONTRARIO(então se não segue a minha condição faça isso)

**Tenha uma boa
Jornada!**

**É hora de você
codar um future
melhor!**

Sobre a autora:



Me chamo Lara T. Bellodi, tenho 36 anos e consegui meu primeiro estágio na área de dados aos 35 anos. Estou cursando meu último ano em Análise e Desenvolvimento de Sistemas e acredito que a área de tecnologia é para todos. A plataforma da DIO foi crucial nessa etapa da minha vida em que eu optei pela mudança de carreira. Acredito que a tecnologia é a resposta para muitos dos nossos problemas e este material é apenas uma referência para te motivar a sempre descobrir mais.

