**Comandos do PostgreSQL - Estados**

**CREATE**

-- Criando o tipo ENUM e a tabela de Estados

-- Um estado possui muitas cidades, logo a PK de estado vai para cidades (1 para N)

CREATE TYPE Regiao AS ENUM ('Norte', 'Nordeste', 'Centro-Oeste', 'Sudeste', 'Sul');

-- Se quiser o BigInt e crie a sua própria sequência, primeiro crie a tabela com o campo id, depois crie a sequencia

-- id BIGINT PRIMARY KEY DEFAULT nextval('estados\_id\_seq')

-- CREATE SEQUENCE estados\_id\_seq;

CREATE TABLE Estados(

  id SERIAL PRIMARY KEY,

  nome VARCHAR(45) NOT NULL UNIQUE,

  sigla VARCHAR(2) NOT NULL UNIQUE,

  regiao Regiao NOT NULL,

  populacao DECIMAL(5, 2) NOT NULL

);

**INSERT**

-- 1 = 1 milhão de população

-- 0.5 = 500 mil de população

-- Inserindo um único registro usando o INSERT INTO

INSERT INTO estados(nome, sigla, regiao, populacao)

VALUES('Acre', 'AC', 'Norte', 0.83);

-- Inserindo o id de sua preferência, pode usar o comando DELETE WHERE id = ?? para excluir depois, caso coloque o id 1000, o próximo INSERT INTO continuar a incrementar de maneira normal

INSERT INTO estados(id, nome, sigla, regiao, populacao)

VALUES(1000, 'Novo', 'NV', 'Sul', 0.0)

-- O id será 30

INSERT INTO estados(nome, sigla, regiao, populacao)

VALUES('Mais novo', 'MV', 'Norte', 0.0)

-- Inserindo vários estados usando o INSERT INTO, colocando sem acento para que se faça um UPDATE nos registros, se quiser mude a população, ou região ou até a sigla.

INSERT INTO estados(nome, sigla, regiao, populacao)

VALUES

  ('Alagoas', 'AL', 'Nordeste', 3.38),

  ('Amapa', 'AP', 'Norte', 0.8),

  ('Amazonas', 'AM', 'Norte', 4.06),

  ('Acre', 'AC', 'Norte', 0.9),

  ('Bahia', 'BA', 'Nordeste', 14.87),

  ('Ceara', 'CE', 'Nordeste', 9.13),

  ('Distrito Federal', 'DF', 'Centro-Oeste', 3.11),

  ('Espirito Santo', 'ES', 'Sudeste', 4.02),

  ('Goias', 'GO', 'Centro-Oeste', 7.02),

  ('Maranhao', 'MA', 'Nordeste', 7.12),

  ('Mato Grosso', 'MT', 'Centro-Oeste', 3.53),

  ('Mato Grosso do Sul', 'MS', 'Centro-Oeste', 2.8),

  ('Minas Gerais', 'MG', 'Sudeste', 21.17),

  ('Para', 'PA', 'Norte', 8.6),

  ('Paraiba', 'PB', 'Nordeste', 4.04),

  ('Parana', 'PR', 'Sul', 11.5),

  ('Pernambuco', 'PE', 'Nordeste', 9.62),

  ('Piaui', 'PI', 'Nordeste', 3.28),

  ('Rio de Janeiro', 'RJ', 'Sudeste', 17.47),

  ('Rio Grande do Norte', 'RN', 'Nordeste', 3.53),

  ('Rio Grande do Sul', 'RS', 'Sul', 11.4),

  ('Rondonia', 'RO', 'Norte', 1.8),

  ('Roraima', 'RR', 'Norte', 0.63),

  ('Santa Catarina', 'SC', 'Sul', 7.28),

  ('Sao Paulo', 'SP', 'Sudeste', 46.29),

  ('Sergipe', 'SE', 'Nordeste', 2.32),

  ('Tocantins', 'TO', 'Norte', 1.57);

**SELECT**

-- Selecionando todos os estados da tabela estados

SELECT \* FROM estados;

-- Filtro com colunas, pegando somente o nome e sigla de cada estado e usando o AS como retorno de cada coluna

SELECT id AS "Id do estado", nome AS "Nome do estado", sigla AS "Sigla do estado" FROM estados;

-- Filtro usando where, retorna os estados da região Sul

SELECT \* FROM estados WHERE regiao = 'Sul';

-- Filtro usando where, retorna os estados que tem a população >= 10 milhões e ordenando de forma crescente, usando o ORDER BY

SELECT \* FROM estados WHERE populacao >= 10 ORDER BY populacao;

-- Filtro usando where, retorna os estados que tem a população >= 10 milhões e ordenando de forma decrescente, usando o ORDER BY e DESC

SELECT \* FROM estados WHERE populacao >= 10 ORDER BY populacao DESC;

**UPDATE**

-- Update com Where, se não usar o where, o comando vai atualizar todos os dados da tabela, coloque um where no id também, pois o id é serial

UPDATE estados SET nome = 'Maranhão' WHERE sigla = 'MA' and id = 29;

-- Update com where, agora vamos atualizar o nome e a população, coloque um where no id também, pois o id é serial

UPDATE estados SET nome = 'Paraná', populacao = 11.5 WHERE sigla = 'PR' AND id = 29;

**DELETE**

-- Excluindo o estado ficticio usando a sigla = MV

DELETE FROM estados WHERE sigla = 'MV';

-- Excluindo o estado ficticio usando o id >= 10

DELETE FROM estados WHERE id >= 1000;

**SELECT com Agregação, Funções, GROUP BY, ORDER BY, DESC BY, SUM, AVG, AS**

-- Retorna a soma de todas as populações

SELECT sum(populacao) as "Total" FROM estados;

-- Retorna a média de todas as populações

SELECT avg(populacao) as "Total" FROM estados;

-- Retorna a região e o Total de cada população por região, agrupando por região, e ordenando de forma crescente

SELECT regiao AS "Região", sum(populacao) as "Total" FROM estados GROUP BY regiao ORDER BY "Total";

-- Retorna a região e o Total de cada população por região, agrupando por região, e ordenando de forma decrescente

SELECT regiao AS "Região", sum(populacao) as "Total" FROM estados GROUP BY regiao ORDER BY "Total" DESC;