Criar e usar um banco de dados

```
CREATE DATABASE NOMEDOBANCO;
USE NOMEDOBANCO;
```

Criar tabelas

```
CREATE TABLE TABELA1(
cod INT PRIMARY KEY,
nome VARCHAR(50)
);

CREATE TABLE TABELA2(
id INT PRIMARY KEY,
nome_prod VARCHAR(50),
preco_prod INT,
cod INT,
FOREIGN KEY(cod)
REFERENCES TABELA1(cod)
);
```

O primary key serve para definir um valor ÚNICO, IMUTÁVEL.
NÃO se utiliza vírgula no último elemento adicionado
Ao fechar o parenteses, USE ponto e vírgula. Se esquecer, você ERROU
O FOREIGN KEY é uma chave que relaciona um elemento em duas ou mais tabelas. Sua sintaxe é: FOREIGN KEY(elemento da tabela que você está) REFERENCES TABELADESEJADA(elemento da tabela desejada.

Inserir valores na tabela

```
#1° modo:
INSERT INTO TABELA1(cod, nome)
VALUES(001, 'APENASUMTESTE');

#2° modo:
INSERT INTO TABELA1
VALUES(001, 'APENASUMTESTE');

INSERT INTO TABELA2
VALUES(1, 'JUSTATEST', 999.99, 001),
      (2, 'JUSTATEST2', 1002.99, 001),
      (3, 'JUSTATEST3', 1049.49, 001);
```

No primeiro modo você especifíca quais campos serão preenchidos com os valores. No modo 2, você, ao omitir os campos, está indicando que TODOS os campos receberão valores, mas fique atento: tais valores deverão ser colocados EM ORDEM conforme a tabela foi criada.

Alterar a tabela

Adicionar coluna

```
ALTER TABLE TABELA1
ADD numero INT;
```

Remover coluna

```
ALTER TABLE TABELA1
DROP COLUMN TABELA1
```

Efetuar consulta

```
SELECT nome_prod FROM TABELA2; #Exibe o elemento especificado

SELECT nome_prod, preco_prod FROM TABELA2; #Exibe os elementos especificados

SELECT * FROM TABELA2; #Exibe TODA a tabela
```

Filtrar dados

O WHERE estabelece um parâmetro de exibição

```
SELECT nome_prod, preco_prod FROM TABELA2
WHERE id = 2;
```

Agrupando dados

Esse comando permite filtrar dados de um determinado grupo.

```
SELECT country, city, count(*) #contar todos
FROM WORLDLIST
GROUP BY country, city;
```

Agrupando dados usando mais funções

O comando abaixo é usado JUNTO com o *group by* e atua como uma condição onde se pode usar agregadores (soma, divisão, maior que, menor que, etc.)

```
SELECT country, city, count(*) #contar todos
FROM WORLDLIST
GROUP BY country, city
HAVING COUNT(*) > 7;
```

Diferença entre *WHERE, GROUP BY* e *HAVING*

Basicamente, o WHERE é usado para filtrar os dados em uma linha, enquanto o GROUP BY filtra os dados em grupos. O HAVING é uma condição usada JUNTO com o group by, definindo mais parâmetros de exibição dos dados de uma tabela.

Exibir os dados que começam ou não por uma letra ou palavra

```
#Que sejam igual a algo:
SELECT * FROM TABELA2
WHERE nome_prod LIKE 'JUSTATEST%'

#Que não sejam iguais a algo:
SELECT * FROM TABELA2
WHERE nome_prod NOT LIKE 'JUSTATEST%'
```

Alterar um valor que já existe no banco de dados

```
UPDATE TABELA2
SET preco_prod = 2179.90
WHERE id = 3;
```

LEMBRE-SE de colocar o WHERE, caso contrário TODOS os valores serão alterados!

Trazer dados de duas ou mais tabelas que possuem relacionamente entre si

```
SELECT  nome, nome_prod, preco_prod #Seleciona elementos de AMBAS as tabelas
FROM TABELA1
INNER JOIN TABELA2
ON TABELA1.cod = TABELA2.cod; #Estabelece o parâmetro de ligação entre elas
```

Retornar os resultados de maneira ordenada

```
SELECT  nome_prod, preco_prod FROM TABELA2
ORDER BY preco_prod; #Maneira crescente

SELECT nome_prod, preco_prod FROM TABELA2
ORDER BY preco_prod desc; #Maneira decrescente
```

Apagar todos os dados de uma vez e uma tabela

```
TRUNCATE TABLE TABELA1;

SELECT * FROM TABELA1;
```

Excluir os dados de uma tabela

```
DELETE FROM TABELA2
WHERE id = 1;

SELECT * FROM TABELA2;
```

Se você não especificar um parâmetro (no caso, o id) você apagará todos os valores da tabela

Apagar uma tabela

```
DROP TABLE TABELA2;
```

Diferença entre *TRUNCATE, DELETE* e *DROP*

- TRUNCATE serve para apagar todas as linhas de uma tabela de uma vez, sem a possibilidade de reversão;
- DELETE permite apagar as linhas conforme o parâmetro especificado (se não houver, apagar-se-ão todas as linhas) e é possível de se reverter;
- DROP apaga a tabela.

Datas

Apresentar data e hora atual

```
SELECT NOW();
```

Apresentar a data atual

```
SELECT CURDATE();
```

Extrair o dia de determinada data

```
#Data atual:
SELECT DAY(CURDATE());

#Qualquer data:
SELECT DAY('2020-12-20') AS Dia;
```

Obs: tu pode chamar isso de tubarão com bigode, se quiser, basta especificar a data com o 'AS algumacoisa'

Extrair o mês de determinada data

```
SELECT MONTH(CURDATE());

SELECT MONTH('2020-12-20') AS Mês;
```

Extrair o ano de determinada data

```
SELECT YEAR(CURDATE());

SELECT YEAR('2020-12-20') AS Ano;
```

Data por extenso

```
SELECT DAY(CURDATE()) AS Dia, MONTH(CURDATE()) AS Mês, YEAR(CURDATE()) AS  Ano;
```

Intervalo com datas

```
#Saber uma data depois de um intervalo de tempo:
SELECT ADDDATE(CURDATE(), INTERVAL 30 DAY);

#Saber a data a partir de hoje (ou de alguma data) até outra data:
SELECT DATEDIFF(NOW(), '2021-01-23');
```