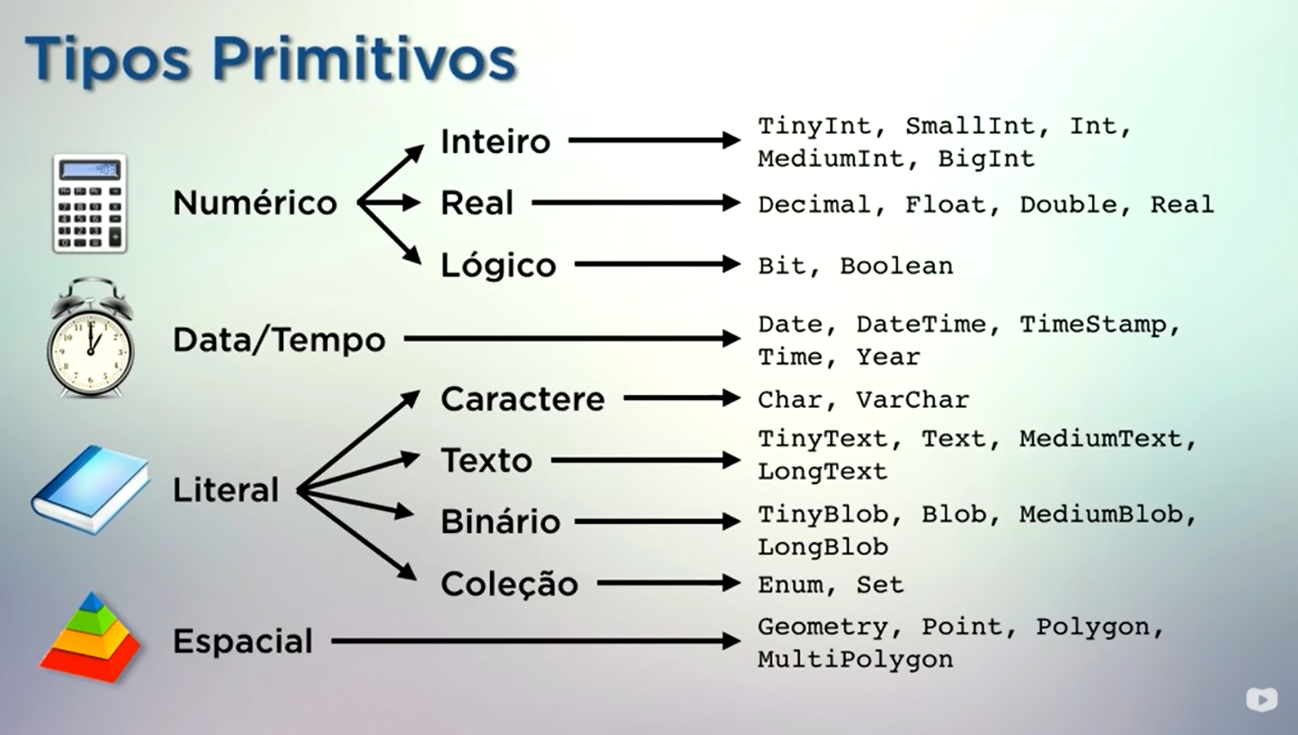
**MySql:**

**Parte 1:**

**Variáveis:**

****

**Entrar no mysql pelo terminal:** Iniciamos ele no xampps e no terminal usamos o i)comando: cd C:\xampp\mysql\bin

ii) comando: mysql -u root -p

iii)senha e vazia

iv) entramos no banco

**Parte 2:**

**Comandos:**

- CREATE DATABASE nomeDoBanco; // criando um banco de dados

- CREATE TABLE nomeTabela(); // criando uma tabela

- drop database nomeDoBanco; // exclui o banco

Show tables // saber quais tabelas temos no nosso banco de dados

**-Criando o banco com caracteres da linguagem brasileira:**

create database cadastro

default character set utf8

default collate utf8\_general\_ci;

**Parte 3:**

Criando tabelas:

**create table pessoas(**

**id int NOT NULL auto\_increment,** // o id recebe um inteiro e não pode deixar de ser incrementado, porem se não incrementado, o “auto\_increment” incrementa automático

**nome varchar(30),** // reserva um espaço de 30 char

**nascimento date,** // data

**sexo enum('M', 'F'),** // enum: so pode digitar “m” ou “F”

**peso decimal(5,2),** // no total 5 casas decimais, sendo 2 depois da virgula. Ex: 123,55

**altura decimal(5,2),** // mesma ideia a cima

**nacionalidade varchar(20) DEFAULT 'Brasil',** // separa 20 char, porem se não for preenchido, e colocado “Brasil” automaticamente

**primary key (id)** // chave primaria onde cada elemento tem a sua, ex: cpf

**) default charset = utf8;** // padrão de escrita do brasil

**Parte 4:**

Inserindo elementos na tabela:

**insert into pessoas**

**values**

**(DEFAULT, 'Ellyza', '2000-05-28', 'F', '60.5', '1.66', 'Brasil'),** // digitar na mesma ordem que os elementos estão na tabela. Id, nome, nascimento ...

**(DEFAULT, 'jean', '1998-02-20', 'M', '80.0', '1.90', DEFAULT),**

**(DEFAULT, 'Pedrinho','1999-03-05','M', '80.5','1.77','EUA');**

**select \* from pessoas;** // mostra o que temos dentro da tabela pessoas

obs: date => ano/mês/dia

**Parte 5:**

Alterando tabela:

**desc pessoas;** // mostra a tabela e suas colunas

**alter table garfanhotos**

**rename to pessoas;** // renomeando tabela

**alter table pessoas**

**add column profissao varchar(10);** // adicionando coluna

**alter table pessoas**

**drop column profissao;** // apagando coluna

**alter table pessoas**

**add column profissao varchar(10) after nome;** // adicionando a coluna profissão depois da “nome”

**alter table pessoas**

**add codigo int first;** // adiciona coluna “código” em primeiro da tabela

**alter table pessoas**

**modify column profissao varchar(20) not null default'';** // modifica coluna

**alter table pessoas**

**change column profissao prof varchar(21);** // altera a coluna

**Parte 6:**

// criando tabela

**create table if not exists cursos(** // se não existe a tabela “curso” crie:

**nome varchar(20) not null unique,**

**descricao text,**

**carga int unsigned,**

**totaulas int unsigned,**

**ano year default '2016'**

**)default charset=utf8;**

**alter table cursos**

**add column idcurso int first;** // adicionando

**alter table cursos**

**add primary key (idcurso);** // adicionando chave primaria em idcurso

**describe cursos;** // mostra a tabela e suas colunas (obs: mesmo que “desc curso”)

**Parte 7:**

**drop table if exists teste;** // apagar tabela teste se ela existir

**Parte 8:**

// alterando as linhas da tabela:

**update cursos** // modifica nome para ‘HTML5’ e o ano para ‘2020’, onde idcurso = 1

**set nome = 'HTML5' , ano = ‘2020’**

**where idcurso = '1'**

**limit 1;** // caso tenha escrito algo errado, me limita alterar apenas uma linha (modo de segurança)

**delete from cursos**

**where idcurso='4'** // deleta uma linha inteira, onde o idcurso=’4’

**limit 1;**

**truncate cursos;** // apaga todas as linhas de cursos

**Parte 9:**

// Comandos SELECT:

//Select parte 1

**select \* from cursos**

**order by nome;** // seleciona todos os cursos por ordem alfabética do nome.

**select nome, ano, carga from cursos**

**order by nome;** // filtra as colunas

**select nome, carga from cursos**

**where ano = ‘2016’**

**order by nome;** // filtrando por ano

podemos usar também com outros relacionais, ex:

“where <= 2015” ou “Where != 2014” ou Where ano > 2016 and carga < 40;

- obs: podemos usar o operador “and” ou “or”

Ex1:

**select nome, ano from cursos**

**where ano between 2014 and 2016;** //entre 2014 e 2016

Ex2:

**select nome, ano from cursos**

**where ano in (2014,2016)** // cursos em 2014 e 2016

**order by ano;**

**Select Parte 2:**

**select \* from cursos**

**where nome like 'P%';** // “p%” escolhe todos com nome que começa com “p”

se “%p” escolhe todos que terminam com “p” e por ultimo “%p%” , escolhe todos que possuem “p”

- obs: % significa que tem alguma coisa ou nada onde ele se encontra

**select \* from cursos**

**where nome like 'ph%p\_';** // escolhe nome onde temos ph + alguma coisa ou nada + ‘p’ + obrigatoriamente algum caractere.

Ou seja, ‘%’ pode ter, como não ter um caractere, ja no ‘\_’ é obrigado a ter.

**select distinct nacionalidade from gafanhotos**

**order by nacionalidade**; // seleciona todas as nacionalidades em ordem, e as que se repetem ele só mostra uma vez

**select count(\*) from cursos**

**order by carga;** // conta quantos cursos temos no banco

**select count(\*) from cursos where carga > '30'**

**order by carga;** // também podemos fazer assim

**select max(carga) from cursos;** // devolve a maior carga

obs: podemos usar “**min**” para o menor

**select sum(totaulas) from cursos where ano = '2016';** // “sum” soma todos os elementos da coluna “totaulas”

**select avg(totaulas) from cursos where ano = '2016';** // “avg” tira a media da coluna

**Select Parte 3:**

// Agrupando elementos de uma coluna

**select nacionalidade, count(nacionalidade) from gafanhotos**

**group by nacionalidade;** // com o ‘group by’ estamos agrupando as nacionalidades, e com o ‘count()’ esta nos informando a quantidade de nacionalidade.

**select ano, count(ano) from cursos**

**group by ano**

**having count(ano) >= 4;** // Aqui selecionamos com o ‘having’ a penas o agrupamento com ano que se repetem 4 ou mais vezes

// Exemplo de Select detro de outro Select:

**select carga, count(\*) from cursos**

**where ano > 2015**

**group by carga**

**having carga > (select avg(carga) from cursos);** // Aqui estaremos mostrando as cargas que estão acima da media geral de cargas

**Parte 10:**

//Modelo Relacional

I)Um para Um -> ex: marido e mulher (um homem tem uma mulher e vice versa)

II)Um para muitos -> um funcionário tem alguns dependentes.

III)Muitos para Muitos -> ex: um produto e clientes (vários clientes podem comprar vários produtos iguais)

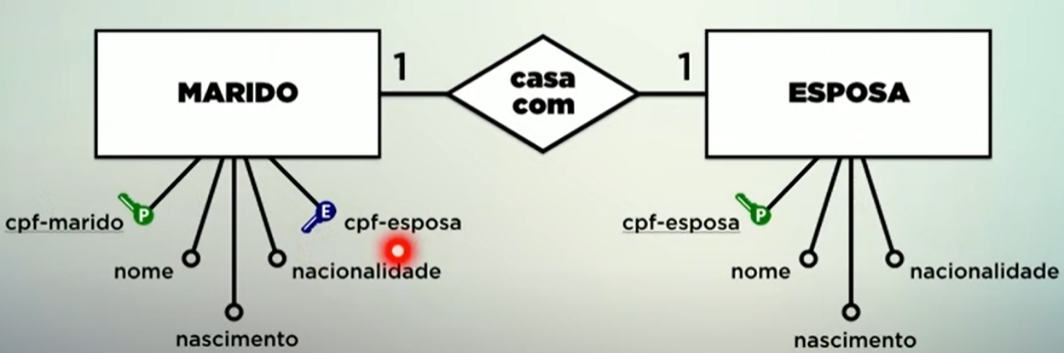
**\*CHAVE ESTRANGEIRA:**

Usada para fazer a relação entre uma entidade e outra.

É basicamente a copia da chave primaria jogada na entidade relacionada.

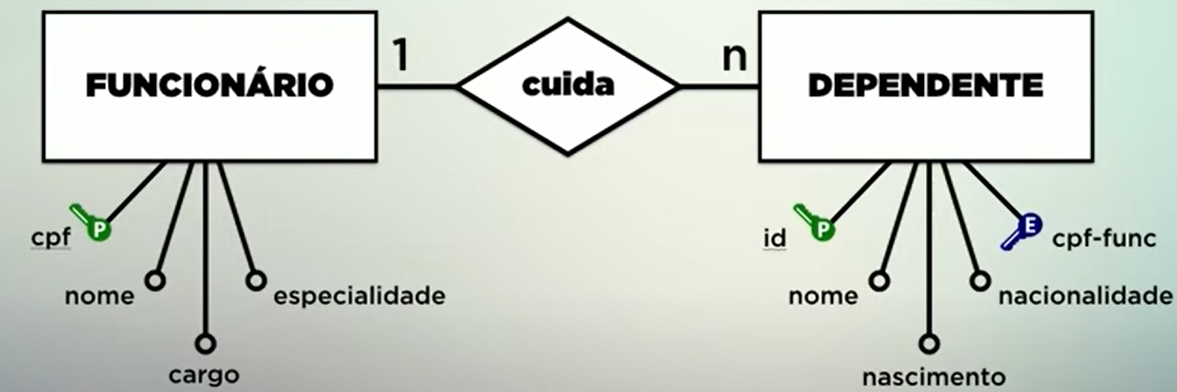
I)Um para Um:

Podemos escolher a entidade dominante e botar a chave estrangeira nela.



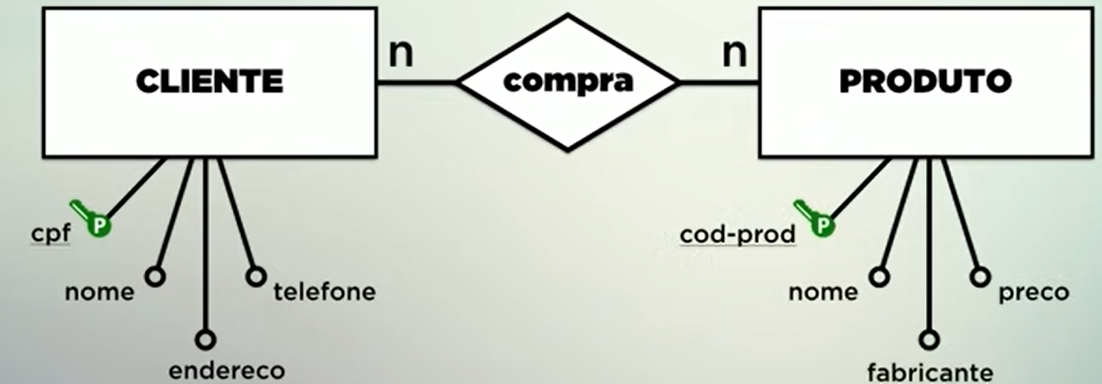
II)Um para Muitos:

Jogamos a chave primaria do lado 1 vira para o lado do muitos:



III)Muitos para Muitos:

O relacionamento vira uma nova entidade: (vamos mostrar essa evolução)

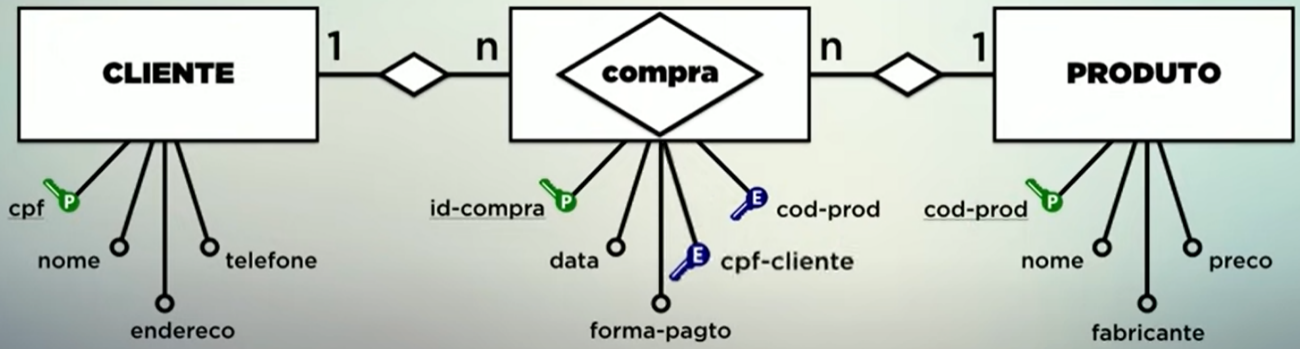


|

|

|

Se transforma em:



PARTE 11:

// Adicionando chaves estrangeiras na pratica:

// primeiro criamos uma coluna nova com o mesmo formaro da que queremos tornar chave estrangeira

**alter table gafanhotos**

**add cursopreferido int;**

// Segundo: transformamos essa nova coluna em chave estrangeira:

**alter table gafanhotos**

**add foreign key (cursopreferido)**

**references cursos(idcurso);**

// adicionando elementos na chave estrangeira:

**update gafanhotos set cursopreferido = '6' where id = '1';**

(obs: você pode usar do workbench para facilitar, adicionando manualmente e depois apertando o botão ‘apply’ no canto inferior direito da tela)

// mostrando elementos da tabela

**select gafanhotos.nome, cursos.nome, cursos.ano from gafanhotos join cursos**

**on cursos.idcurso = gafanhotos.cursopreferido;**

obs(o ‘join’ esta juntando gafanhotos e cursos, e o ‘on’ esta relacionando a chave primaria e a estrangeira)

// podemos usar com apelido também, deixando o código menor:

**select g.nome, c.nome, c.ano**

**from gafanhotos as g join cursos as c**

**on c.idcurso = g.cursopreferido;**

// mostrando todos os cursos, até os que não são preferidos por ninguém:

**select g.nome, c.nome, c.ano**

**from gafanhotos as g right outer join cursos as c**

**on c.idcurso = g.cursopreferido;**

right outer -> seleciona toda a tabela da direita. Se fosse left, selecionaria a da esquerda

PARTE 12:

// Adicionando chaves estrangeiras Muitos para Muitos

// Nesse caso criamos uma nova tabela para interligar os 2.

// No exemplo a seguir estamos relacionando os gafanhotos aos cursos:

**create table gafanhoto\_assiste\_curso(**

**id int NOT NULL AUTO\_INCREMENT,**

**data date,**

**idgafanhoto int,**

**idcurso int,**

**PRIMARY KEY (id),**

**FOREIGN KEY(idgafanhoto) REFERENCES gafanhotos(id),**

**FOREIGN KEY(idcurso) REFERENCES cursos(idcurso)**

**) default charset = utf8;**

// criamos a tabela gafanhoto\_assiste\_curso para relacionar cursos e gafanhotos

// Logo depois vamos preenchendo ela:

**insert into gafanhoto\_assiste\_curso values**

**(default, '2014-01-03', '1','2');**

// E para interligar e imprimir usamos 2 ‘join’ :

**select g.nome, c.nome from gafanhotos g**

**join gafanhoto\_assiste\_curso a**

**on g.id = a.idgafanhoto**

**join cursos c**

**on c.idcurso = a.idcurso**

**order by g.nome;**