PROGRAMADOR

BANCO DE DADOS MySQL

CONTEÚDO DO MÓDULO

1. Introdução a banco de dados
2. Estrutura de banco de dados
3. Manipulação de dados
4. Banco de dados com Java

Conteúdo Aula 1

1 Introdução a banco de dados

2 Estrutura de banco de dados

Introdução a banco de dados

O que são dados?

São informações que possuem algum valor para o usuário, portanto necessitam ser armazenados em algum local que possam ser recuperados posteriormente.

O que é um banco de dados?

São coleções de dados que se relacionam de forma a criar algum sentido e dar mais eficiência durante uma pesquisa.

O banco de dados mantém as informações em um SGBD (Sistema de gerenciamento de banco de dados).

O termo banco de dados é utilizado como sinônimo de SGBD.

Introdução a banco de dados

Existem alguns modelo de banco de dados:

- **Hierárquico**
- **Relacional**
- **Orientado a Objetos**

Modelo relacional

O modo relacional representa os dados como uma coleção de tabelas (relações).

Todas as tabelas possuirão um nome, e um conjuntos de atributos com um tipo de dado definido.

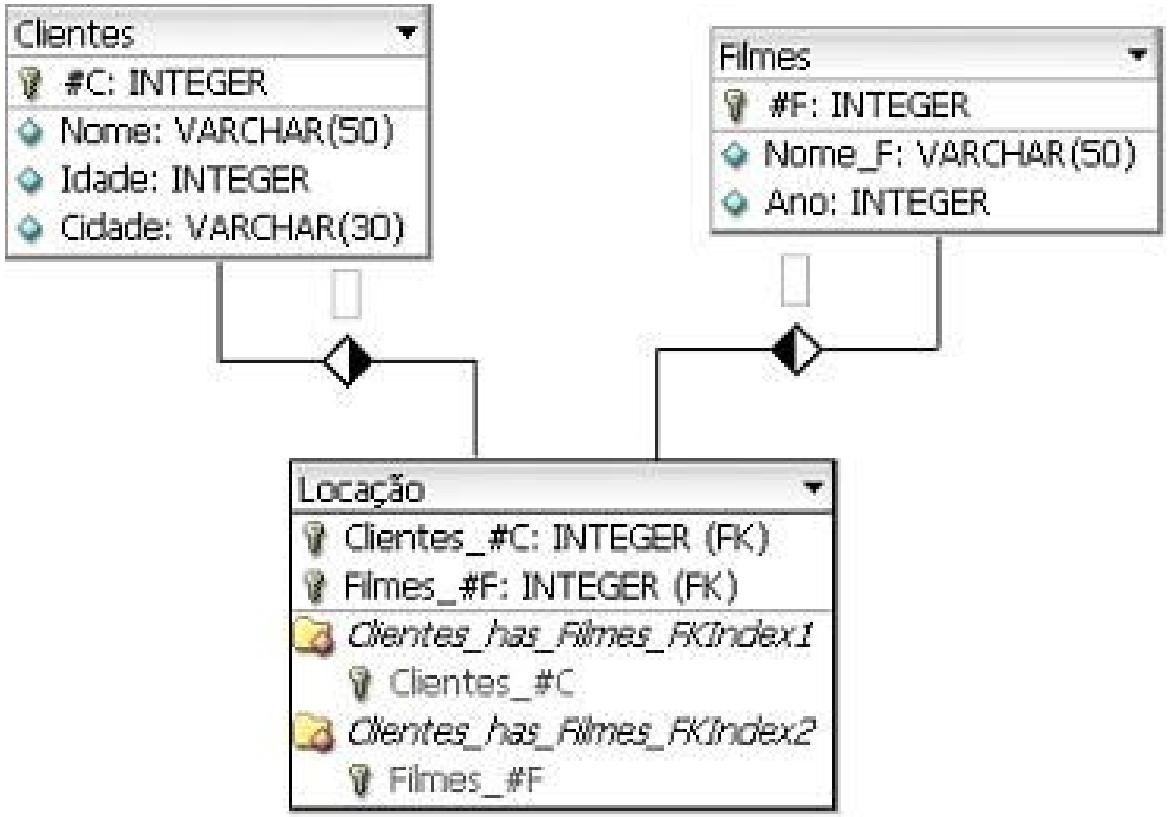
Cada coluna possuirá dados de um mesmo tipo.

Cada linha é chamada de tupla.
O nome de uma coluna é chamado de atributo ou campo.

Modelo relacional

O modelo relacional trabalha com entidades e relacionamentos.
"Uma entidade é uma 'coisa' ou um 'objeto' no mundo real que pode ser identificada de forma unívoca em relação a todos os outros objetos"
(SILBERSCHATZ; KORTH; SUDARSHAN, 1999, p. 21).

Modelo relacional



Modelo relacional

Cada item apresentado no modelo corresponde a uma coluna no banco de dados.
Cada atributo deve possuir valor único em cada tabela.

Os nomes das tabelas e colunas devem ser sempre o mais claro possível. (Tabela: funcionarios, Campo: matricula, nome, endereco, cargo, salario).

Exemplo de uma tabela

funcionarios				
matricula	nome	endereco	cargo	salario
0001	Maria	Rua João Augusto	Programador	4000
0002	João	Rua das Nações	Analista	5000
0004	Joana	Rua Tiradentes	Arquiteto	6000

Conteúdo Aula 2

Estrutura de banco de dados 1

Manipulação de dados 2

Estrutura de banco de dados - Chave primária PK

Chave primária: atributo ou combinação de atributos, capaz de identificar de forma única uma linha de uma tabela.

Podemos ter a uma chave primária candidata: identificador único que garante que nenhuma outra tupla seja igual. (Identificador já presente na tabela)

Nem sempre temos uma chave candidata na tabela, por esse motivo criamos uma chave primária para a nossa tabela.

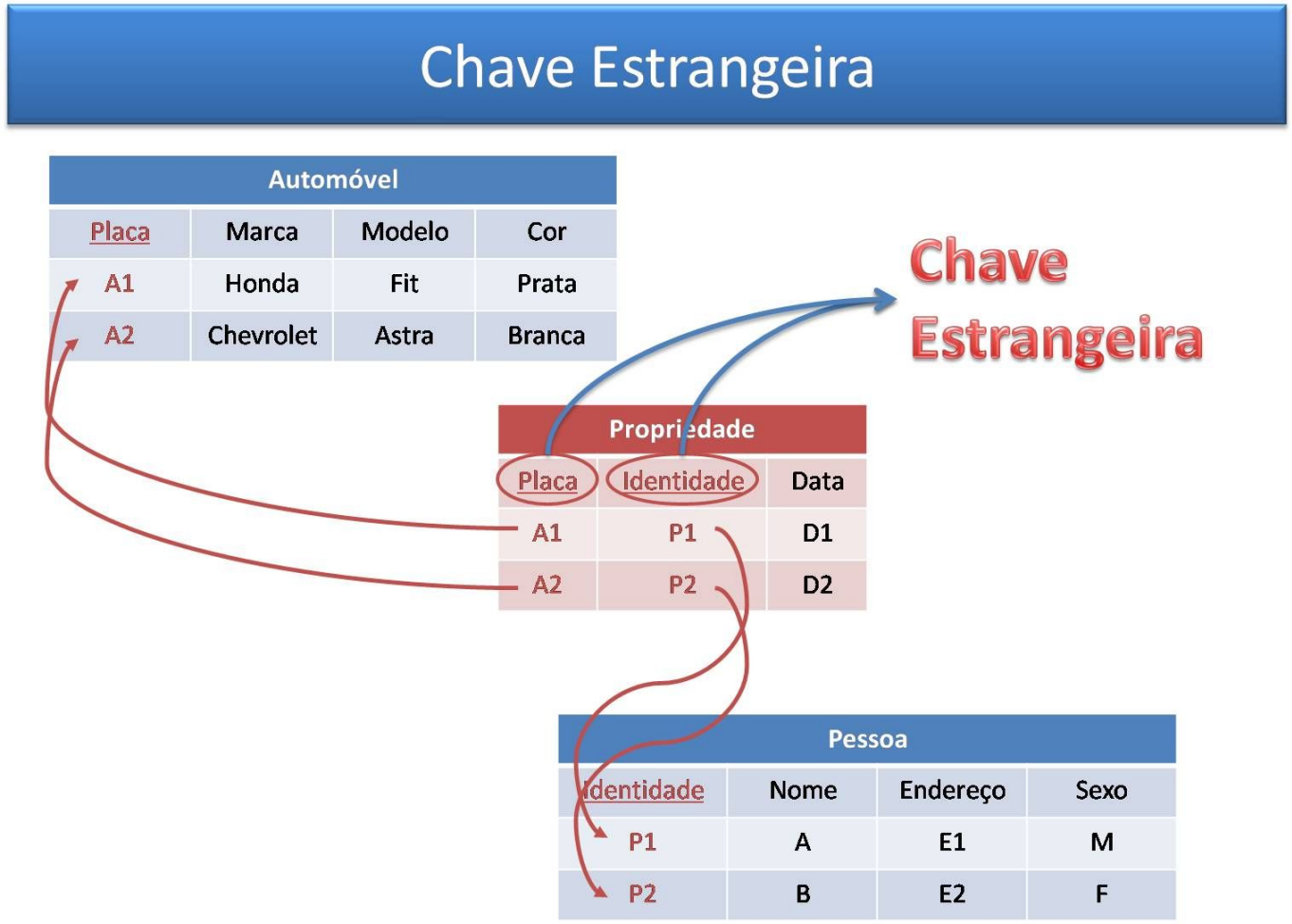
Estrutura de banco de dados - Chave estrangeira FK

Uma chave estrangeira é um campo, que aponta para a chave primária de outra tabela ou da mesma tabela.

Ou seja, passa a existir uma relação entre duplas de duas tabelas ou de uma única tabela.

A finalidade da chave estrangeira é garantir a integridade dos dados referenciais, pois apenas serão permitidos valores que supostamente vão aparecer na base de dados.

Estrutura de banco de dados - Chave estrangeira FK



Estrutura de banco de dados

Cada banco de dados (Oracle, Postgres, MySQL, SQL Server, etc), possui seus próprios tipos de dados, dessa forma devemos conhecer o banco de dados que iremos trabalhar.

As características básicas dos tipos de dados que são comuns a todos os SGBD.

Numéricos (Integer, Real): normalmente permitem definirmos os números de casas decimais.

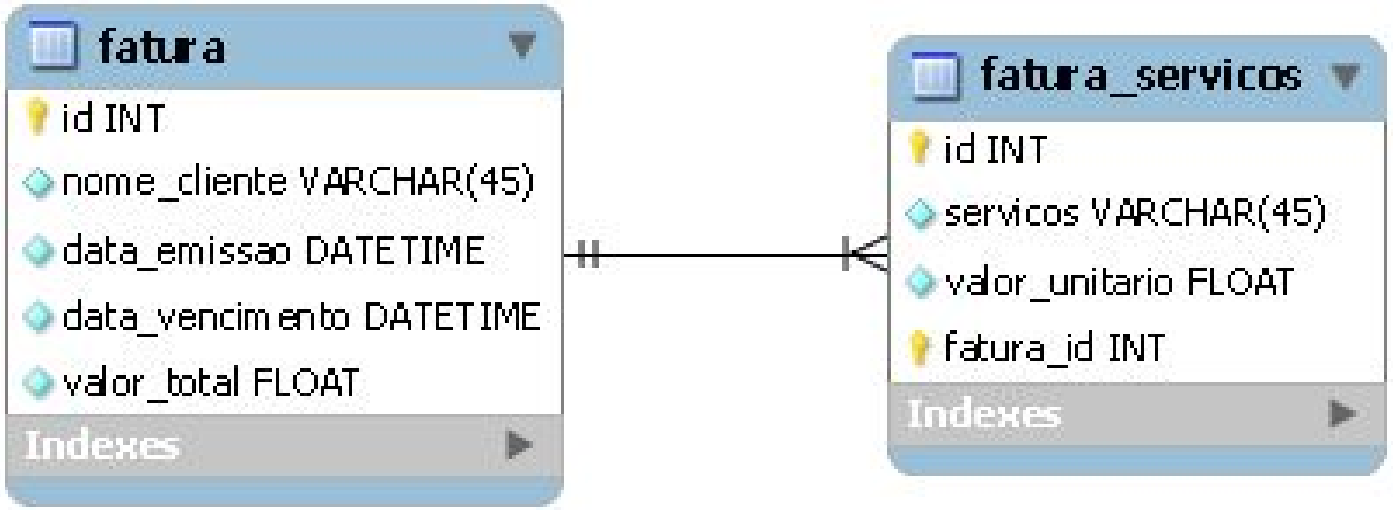
Caracteres (Varchar): normalmente permite a definição do número máximo de caracteres.

Datas (Date).

Estrutura de banco de dados

NULL: não é um tipo, porém representa a ausência de valores. Quando criamos um campo é possível definirmos se o campo é de preenchimento obrigatório, não aceita valor null.

Estrutura de banco de dados

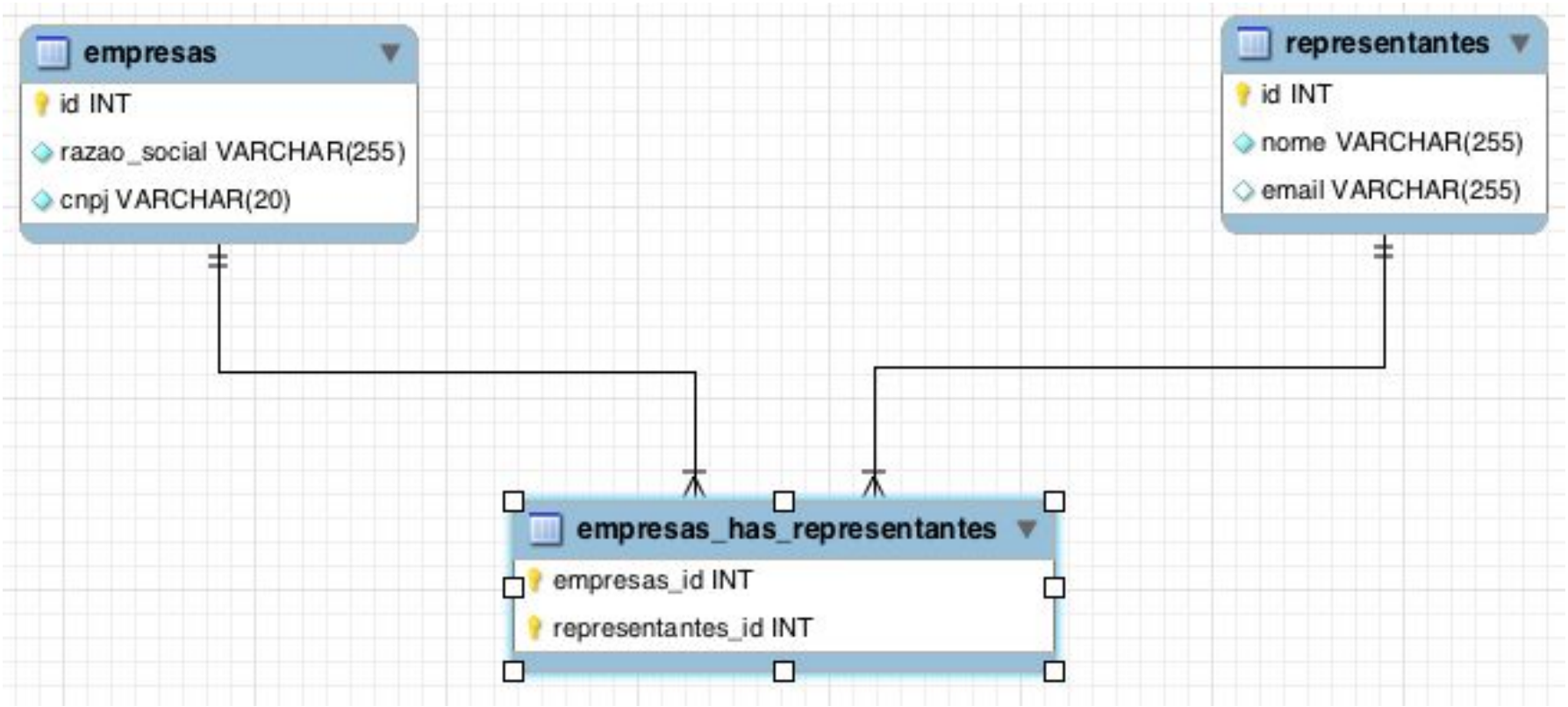


Relacionamento

"Um relacionamento é uma associação entre uma ou várias entidades" **(SILBERSCHATZ; KORTH; SUDARSHAN, 1999, p. 24).**

"Um conjunto de relacionamentos é um conjunto de relacionamentos do mesmo tipo" **(SILBERSCHATZ; KORTH; SUDARSHAN, 1999, p. 25).**

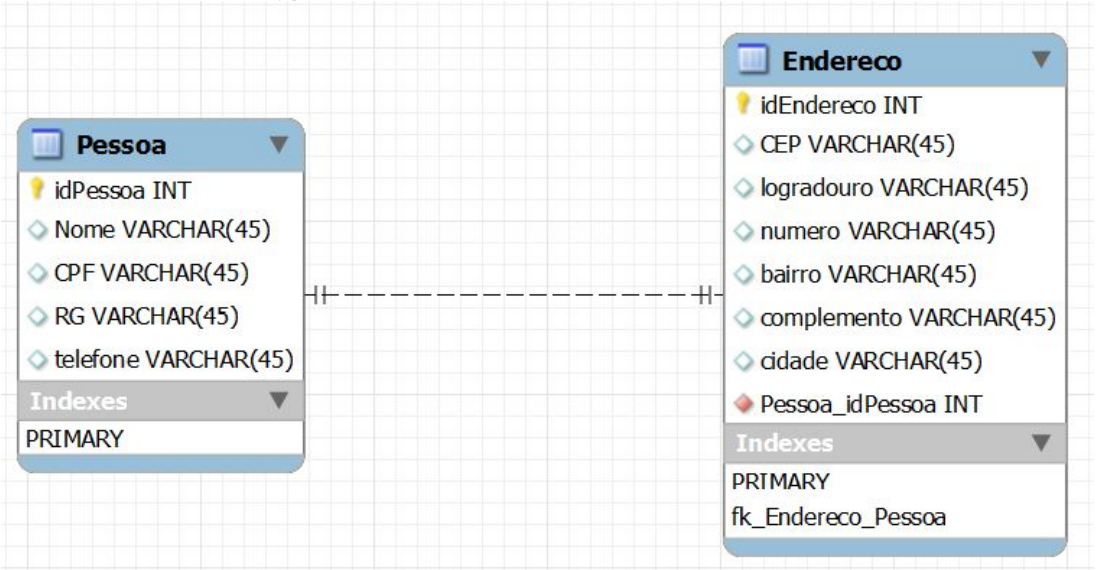
Relacionamento



Relacionamento

Um para um

Nesse relacionamento cada registro tem apenas 1 ligação com 1 registro da outra tabela. 1 pessoa possui apenas 1 endereço e cada endereço é único de uma pessoa.



Relacionamento

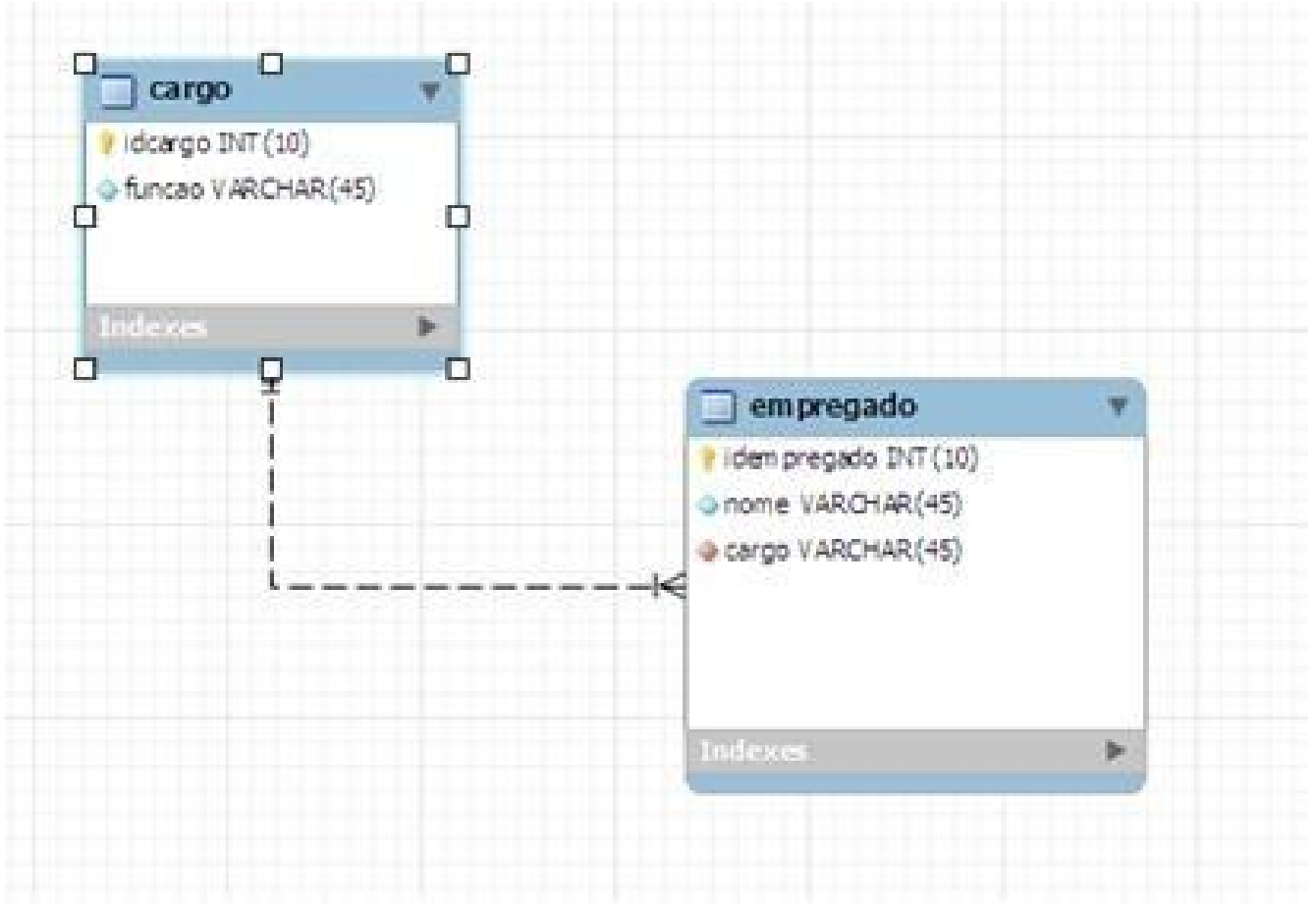
Um para muitos

Nesse relacionamento cada registro da tabela cargo pode estar associado a vários registros da tabela empregado. Enquanto cada registro da tabela empregado por estar associado a apenas um registro da tabela cargo.

O que significa que em uma empresa 1 empregado pode ter apenas 1 cargos, porém, pode existir cargos com mais de 1 funcionário.

Relacionamento

Um para muitos



Relacionamento

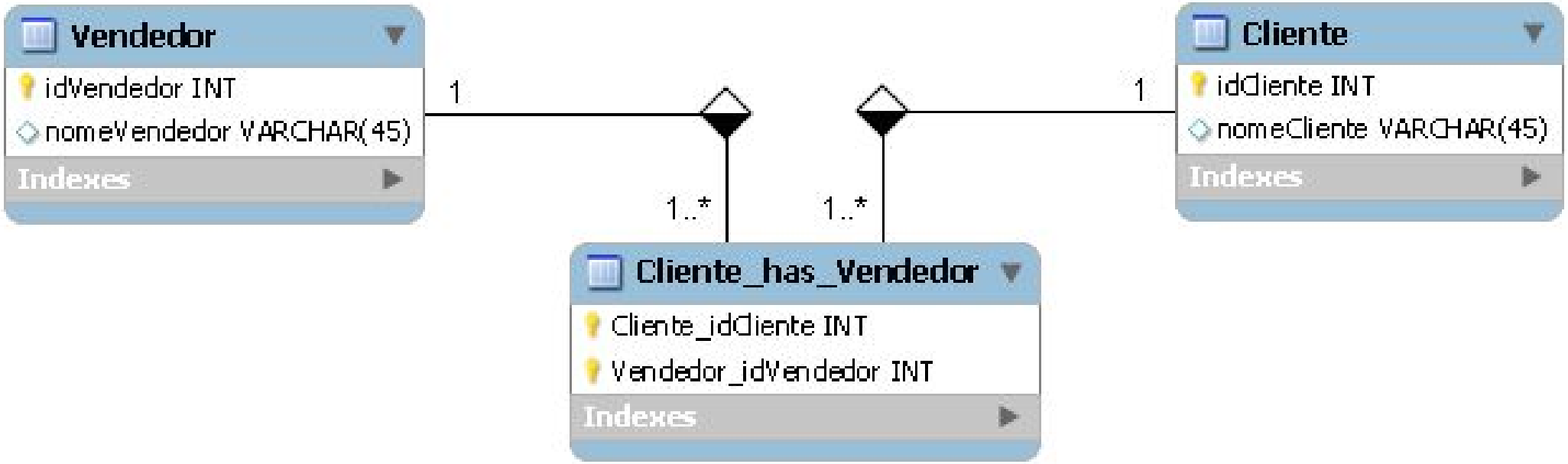
Muitos para muitos

Nesse relacionamento um registro da tabela vendedor poderá estar associado a vários registros da tabela cliente, e um registro da tabela cliente poderá estar associado a vários registros da tabela vendedor.

O que significa que em 1 vendedor poderá vender para farios clientes e 1 cliente poderá comprar de vários vendedores.

Relacionamento

Muitos para muitos



Linguagem SQL

Structured Query Language, ou Linguagem de Consulta Estruturada ou SQL, é a linguagem de pesquisa declarativa padrão para banco de dados relacional.

SQL nos permite diversas operações:

- **Criar tabela.**
- **Inserir dados.**
- **Atualizar dados.**
- **Consultar dados**
- **Remover dados.**

Comandos DDL

DDL (Data Definition Language) é utilizado para manutenção das estruturas de dados.

CREATE TABLE: cria de tabelas.

ALTER TABLE: altera a estrutura das tabelas, adicionar coluna, remover coluna, alterar coluna.

DROP TABLE: remove tabelas.

Criando tabelas

Precisamos definir alguns detalhes antes de criar nossas tabelas:

Quais serão os nomes das tabelas.

Qual será a chave primária.

Quais nomes e tipos de dados serão atribuídos a cada coluna.

Quais as colunas serão obrigatórias.

Criando tabelas

CREATE TABLE nomeTabela (listaCampos [definicaoChaveEstrangeira]);

listaCampos: é uma lista de campos separados por vírgula seguindo a seguinte sintaxe:

nomeCampo tipoCampo [NOT NULL] [PRIMARY KEY]

NOT NULL: campos de preenchimento obrigatório.

PRIMARY KEY: campos que formam a chave primária.

Criando tabelas

Quando precisarmos criar as chaves estrangeiras devemos seguir a seguinte sintaxe após a definição da lista de campos:

CONSTRAINT FOREING KEY (nomeColuna) REFERENCES tabelaPrimaria (nomeChavePrimaria)

nomeColuna: coluna que será utilizada como chave estrangeira.

tabelaPrimaria: nome da tabela que terá o relacionamento.

nomeChavePrimaria: nome da chave primária na tabela primária.

Criando tabelas

```
create table usuario (
    id integer not null primary key auto_increment,
    nome varchar(50) not null,
    endereco varchar(50) not null,
    perfil integer not null,
    login varchar(20) not null,
    senha varchar(20) not null
) ENGINE=InnoDB;
```

Criando tabelas

```
create table livro (
    id integer not null primary key auto_increment,
    nome varchar(50) not null,
    autor varchar(50) not null,
    ano_edicao integer not null
) ENGINE=InnoDB;
```

Criando tabelas

```
create table emprestimo (
    id integer not null primary key auto_increment,
    id_usuario integer not null,
    id_livro integer not null,
    constraint foreign key (id_usuario) references usuario
(id),
    constraint foreign key (id_livro) references livro (id)
) ENGINE=InnoDB;
```

Exercício

Criar as tabelas do projeto no banco. Tabelas essas que existem os exemplos no slides passados.

Alterando tabela

ALTER TABLE nomeTabela MODIFY COLUMN nomeColuna tipoDado [NULL|NOT NULL]

Alterando o número máximo de caracteres da coluna nome da tabela usuario:

ALTER TABLE usuario MODIFY COLUMN nome VARCHAR(30)

Alterando tabela

Alterando a coluna endereco da tabela usuario para que seja de preenchimento opcional:

```
ALTER TABLE usuario MODIFY COLUMN
endereco VARCHAR(30) NULL
```

Alterando tabela

Adicionando coluna nas tabelas.

ALTER TABLE nomeTabela ADD COLUMN nomeColuna tipoDado[NULL|NOT NULL]

Adicionando a coluna cpf na tabela usuario:

ALTER TABLE usuario ADD COLUMN cpf VARCHAR(30)

Alterando tabela

Removendo colunas.

ALTER TABLE nomeTabela DROP COLUMN nomeColuna

Removendo a coluna cpf da tabela usuario:

ALTER TABLE usuario DROP COLUMN cpf

Exercício

1. **Adicionar coluna idade campo obrigatório na tabela usuário.**
2. **Alterar coluna idade para não ser um campo obrigatório.**
3. **Remover coluna idade.**

Renomeando colunas

Renomeando colunas.

ALTER TABLE nomeTabela CHANGE COLUMN nomeAntigo nomeNovo tipoDado [NULL|NOT NULL]

Renomeando a coluna nome da tabela usuario para tx_nome:

ALTER TABLE usuario CHANGE COLUMN nome tx_nome VARCHAR(50) NOT NULL

Renomeando tabela

Renomeando tabela.

ALTER TABLE nomeAntigo RENAME [TO] novoNome

Renomeando a tabela usuario para usuarios:

ALTER TABLE usuario RENAME usuarios

Outra opção para renomear tabelas:

RENAME TABLE usuario TO usuarios

Removendo tabela

Removendo tabela.

DROP TABLE nomeTabela

Removendo a tabela usuario:

DROP TABLE usuario

Conteúdo Aula 3

Manipulação de dados 1

Comandos DML

DML (Data Manipulation Language) é utilizado para manipular os dados nas tabelas.

INSERT: comando para adicionar registros.

UPDATE: comando para atualizar valores nas tabelas.

DELETE: comando para remover registros.

Inserindo dados

INSERT INTO nomeTabela [(listaColunas)] VALUES (listaValores)

listaColunas: nome dos campos da tabela separados por vírgula.

listaValores: os valores devem seguir a mesma ordem das colunas, separados por vírgula. Quando o valor for um texto ou data, ele deve ser escrito entre aspas simples (‘). Valores numéricos e NULL não precisam das aspas.

Inserindo dados

Inserindo registro na tabela usuário, com nome = 'Maria', endereco = 'Rua das Nações', Perfil = 1, login = 'maria' e senha = '123456'

INSERT INTO usuario(nome, endereco, perfil, login, senha) VALUES ('Maria', 'Rua das Nações', 1, 'maria', '123456');

Exercício

1. **Inserir registros na tabela usuario.**
2. **Inserir registros na tabela livro.**
3. **Inserir registros na tabela emprestimo.**

Atualizando dados

UPDATE nomeTabela SET listaAtualizacoes [WHERE condicao]

listaAtualizacoes: lista de campos com as alterações dos valores separados por vírgula seguindo a sintaxe a seguir.

nomeColuna = novoValor

WHERE: funciona como um filtro, caso não seja utilizado, a atualização se aplicará a todos os registros daquela tabela.

Atualizando dados

Atualizando o registro da tabela usuário que possui nome Maria para nome = 'Joana' e login = 'joana'.

UPDATE usuario set nome = 'Joana', login = 'joana' WHERE nome like 'Maria';

Exercício

1. **Atualizar valores na tabela usuario.**
2. **Atualizar valores na tabela livro.**
3. **Atualizar valores na tabela emprestimo.**

Removendo dados

DELETE FROM nomeTabela [WHERE condicao]

WHERE: funciona como um filtro, caso não seja utilizado, a atualização se aplicará a todos os registros daquela tabela.

Removendo o registro da tabela usuário que possui nome = Maria.

DELETE FROM usuario WHERE nome like 'Maria'

Exercício

1. **Remover registro na tabela usuario.**
2. **Remover registro na tabela livro.**
3. **Remover registro na tabela emprestimo.**

Comandos DQL

DQL (Data Query Language) é utilizado para realizar as consultas dos dados em nossas bases de dados.

SELECT: comando para as consultas dos dados nas tabelas.

Select - Consultas básicas

Antes de realizarmos uma consulta no banco de dados, necessitamos saber quais as colunas que desejamos saber, e quais as tabelas.

SELECT listaCampos FROM listaTabelas

listaCampos: lista separada por vírgula com os campos que necessitamos. Também podemos utilizar o (*) que funciona como um coringa, trazendo todos os campos da tabela.

listaTabelas: lista separada por vírgula com os nomes das tabelas.

Select - Consultas básicas

Listar todos os registros da tabela usuario:

SELECT * FROM usuario

Listar o login e senha de todos os registros da tabela usuario:

SELECT login, senha FROM usuario

Select - Filtrando conteúdo

Como podemos fazer para trazer apenas os livro de ano maior do que 2010?

Precisamos utilizar filtros nas consultas.

SELECT listaCampos FROM listaTabelas WHERE condicao

condicao: expressão que retorna um valor boolean. Por exemplo ano_edicao > 2010.

Select - Filtrando conteúdo

Listar todos os livros que o ano de edição seja maior do que 2010.

SELECT * FROM livro WHERE ano_edicao > 2010

Listar todos os usuários que possuem senha ‘123456’.

SELECT * FROM usuario WHERE senha like ‘123456’

Select - Operadores

Operadores aritméticos:

- **Adição (+)**
- **Subtração (-)**
- **Multiplicação (*)**
- **Divisão(/)**

Select - Operadores

Operadores relacionais ou de comparação:

- **Igualdade (=)**
- **Desigualdade (<> ou !=)**
- **Maior que (>)**
- **Menor que (<)**
- **Maior ou igual (>=)**
- **Menor ou igual (<=)**

Select - Operadores

Operadores lógicos:

e (AND)

ou (OR)

negação (NOT)

Select - Operadores

SELECT * FROM usuario WHERE nome = 'Maria' AND login = 'maria'

SELECT * FROM usuario WHERE nome = 'Maria' OR login = 'joana'

Select - Operador IS NULL

O operador IS NULL é utilizado para comparar se um campo possui o valor NULL.

Listar todos os usuários cujo endereço não foi preenchido.

```
SELECT * FROM usuario WHERE endereco IS NULL
```

Select - Operador IS NULL

Listar todos os usuários cujo endereço foi preenchido.

SELECT * FROM usuario WHERE endereco IS NOT NULL

Select - Operador LIKE

Like é utilizado para comparar valores texto, podendo utilizar campo coringas.

% - representa 0 ou mais caracteres.

_ - representa um único caractere.

SELECT * FROM nomeTabela WHERE campo LIKE '%Teste'

Select - Operador LIKE

Listar todos os usuário cujo nome comece com ‘Ma’.

SELECT * FROM usuario WHERE nome LIKE ‘Ma%’.

Listar todos os usuário cujo nome possua a primeira letra qualquer valor, porém o restante seja ‘aria’.

SELECT * FROM usuario WHERE nome LIKE ‘_aria’.

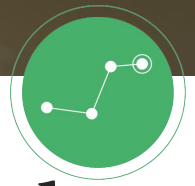
Exercício

1. **Consultar todos os usuário que comecem com a letra ‘A’.**
2. **Consultar todos os livros cujo ano de edição seja maior do que 2000.**
3. **Consultar todos os livros cujo autor tenho o nome “José”.**

Ordenação

Quando necessitarmos retornar a lista de registros ordenados por determinado campo, podemos utilizar a cláusula ORDER BY.

SELECT listaCampos FROM listaTabelas [WHERE condicao] ORDER BY listaCampos [ASC|DESC]



Ordenação

Listar todos os usuário ordenados pelo nome.

SELECT * FROM usuario ORDER BY nome;

Listar todos os usuário ordenados pelo nome em ordem decrescente.

SELECT * FROM usuario ORDER BY nome DESC;

Exercício

1. **Consultar todos os usuário do sistema ordenados pelo login.**
2. **Consultar todos os livros ordenados pelo ano de edição.**

Comandos DCL

DCL (Data Control Language) é utilizado para realizar os controles de permissões de acesso ao banco de dados.

GRANT: comando para adicionar permissões a um usuário do banco de dados.

REVOKE: comando para remover permissões de um usuário do banco de dados.

Consultando várias tabelas

Para realizarmos consultas a várias tabelas em um único SQL podemos utlizar os JOINs.

Utilizando Join

Exercício

1. **Consultar todos os livros que um determinado usuário possui empréstimo.**

Conteúdo Aula 4

Banco de dados com Java 1