Banco de Dados

O que é a

- Coleção de dados logicamente coerente
- Criados e mantidos por SGBDs

Persistência

- Processo de armazenamento e captura de dados

Tabelas

- Forma como os dados estão relacionados

SQL

- Linguagem de pesquisa declarativa padrão para bancos de dados relacionais

Atributos

Correspondem a cada coluna da tabela

SGBD

Base de Dados

Dados armazenados

Metadados

Descreve a forma como os dados são armazenados

Interface

Para que seus clientes possam incluir, alterar ou consultar dados

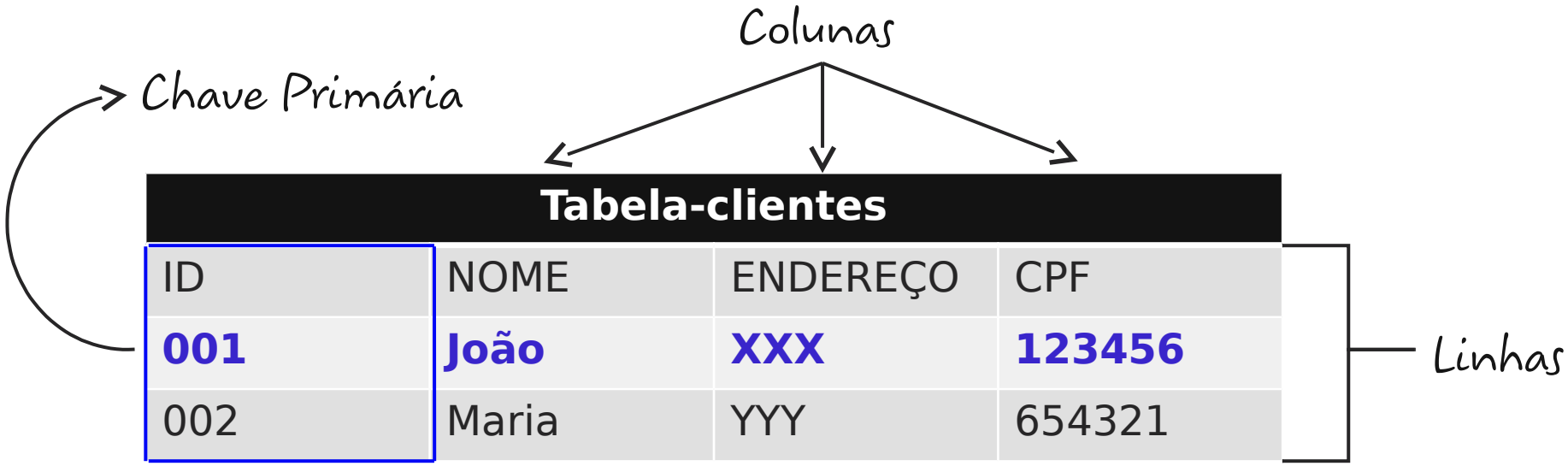
Integridade

Garantir a integridade dos dados

Segurança

Controle de acesso

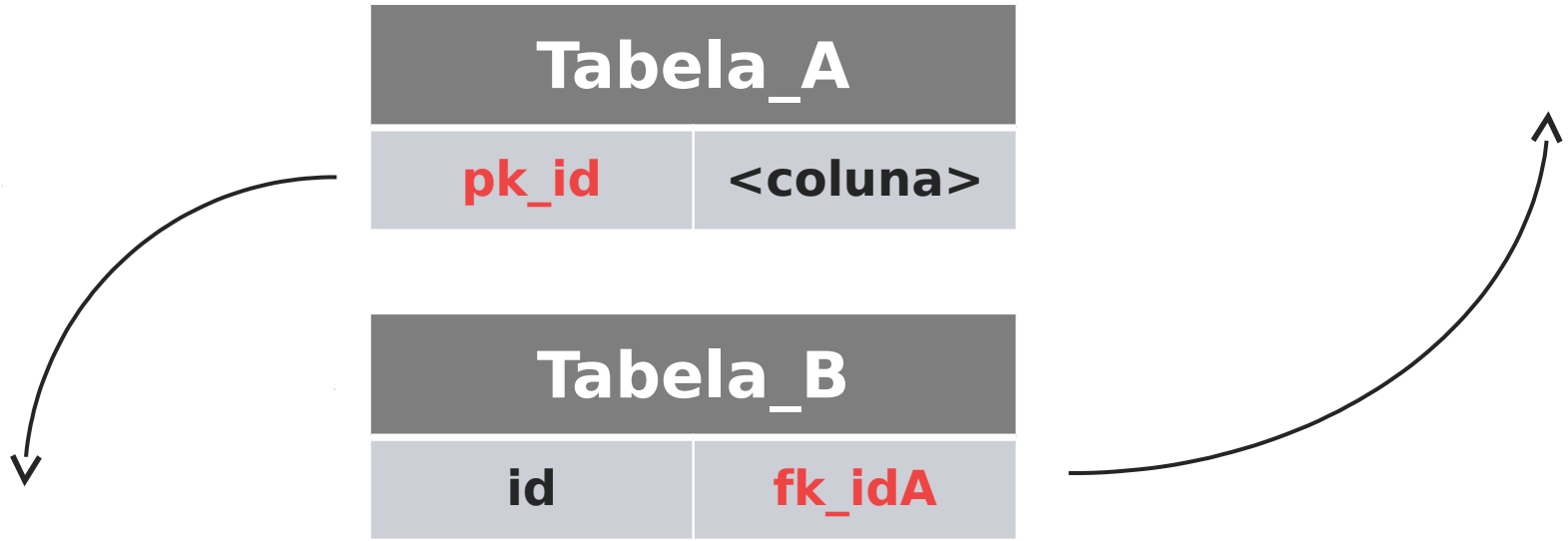
Estrutura de uma Tabela



Relacionamento

Chave Estrangeira (Foreign Key)

Estabelece relacionamento entre duas tabelas
Corresponde à chave primária de outra tabela



Chave Primária (Primary Key)

Identifica de forma única uma linha
da tabela

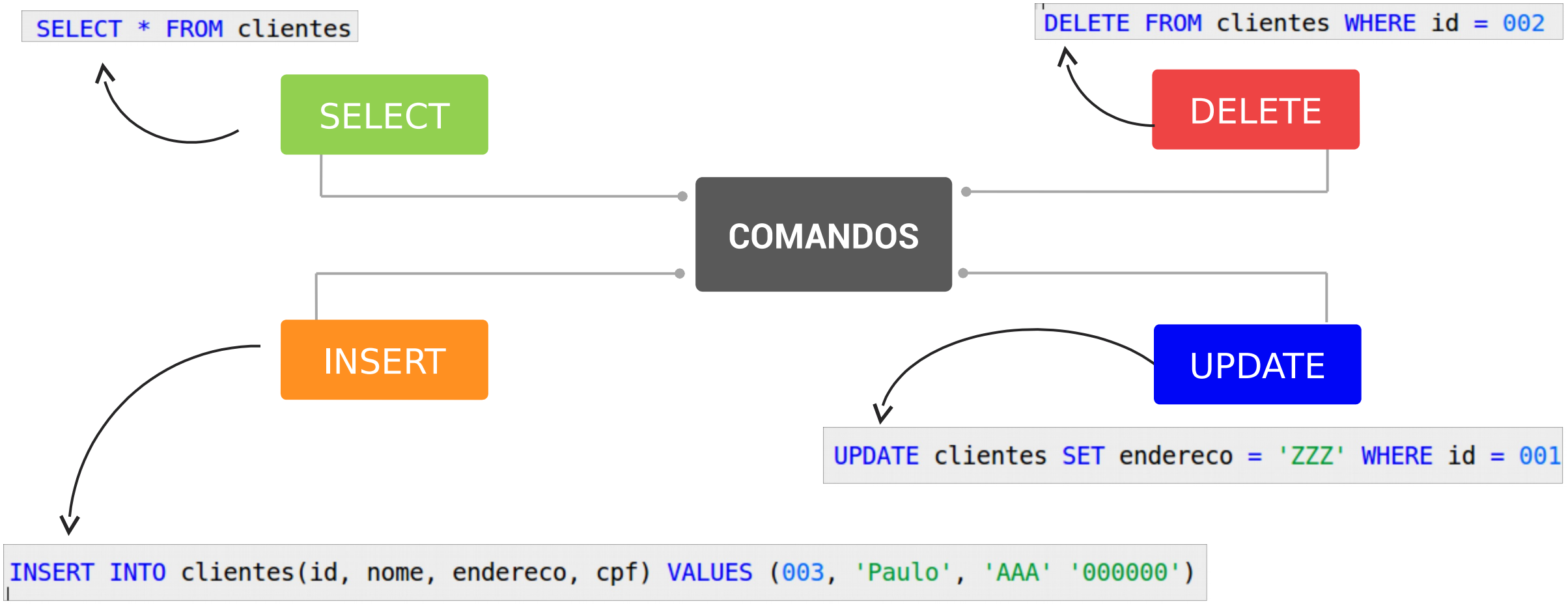
SQL

DDL	DCL	DTL	DML
Create Drop Filter	Grant Revoke	Commit Rollback	Select Insert Update Delete

SQL

Comandos	Cláusulas	**Operadores Lógicos**	Operadores Relacionais
SELECT INSERT UPDATE DELETE	FROM WHERE GROUP BY ORDER BY Outras...	AND OR NOT	< > >= <= = <>

Comandos


SELECT * FROM clientes
DELETE FROM clientes WHERE id = 002
SELECT
DELETE
COMANDOS
INSERT
UPDATE
UPDATE clientes SET endereco = 'ZZZ' WHERE id = 001
INSERT INTO clientes(id, nome, endereco, cpf) VALUES (003, 'Paulo', 'AAA' '000000')

Comandos - SELECT

Consulta de Registro

```
SELECT * FROM clientes WHERE id = 001
```

Coluna "id"

Tabela-clientes			
ID	NOME	ENDEREÇO	CPF
001	**João**	**XXX**	**123456**
002	Maria	YYY	654321

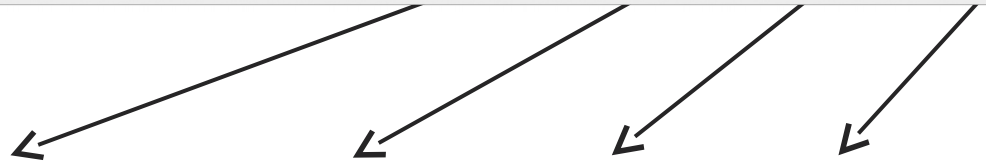
Resultado do SELECT

ID	NOME	ENDEREÇO	CPF
001	**João**	**XXX**	**123456**

Comandos - INSERT

Inserção de Registro

```
INSERT INTO clientes(id, nome, endereco, cpf) VALUES (003, 'Paulo', 'AAA' '000000')
```



ID	NOME	ENDEREÇO	CPF
003	Paulo	AAA	000000

Resultado do SELECT

Tabela-clientes			
ID	NOME	ENDEREÇO	CPF
001	João	XXX	123456
002	Maria	YYY	654321
003	**Paulo**	**AAA**	**000000**

Comandos - UPDATE

```
UPDATE clientes SET endereco = 'ZZZ' WHERE id = 001
```

Alteração de Registro

Coluna "id"

Tabela-clientes			
ID	NOME	ENDEREÇO	CPF
001	João	XXX	123456
002	Maria	YYY	654321
003	Paulo	AAA	000000

Resultado do UPDATE

Tabela-clientes			
ID	NOME	ENDEREÇO	CPF
001	**João**	**ZZZ**	**123456**
002	Maria	YYY	654321
003	Paulo	AAA	000000

Comandos – DELETE

Remoção de registros

```
DELETE FROM clientes WHERE id = 002
```

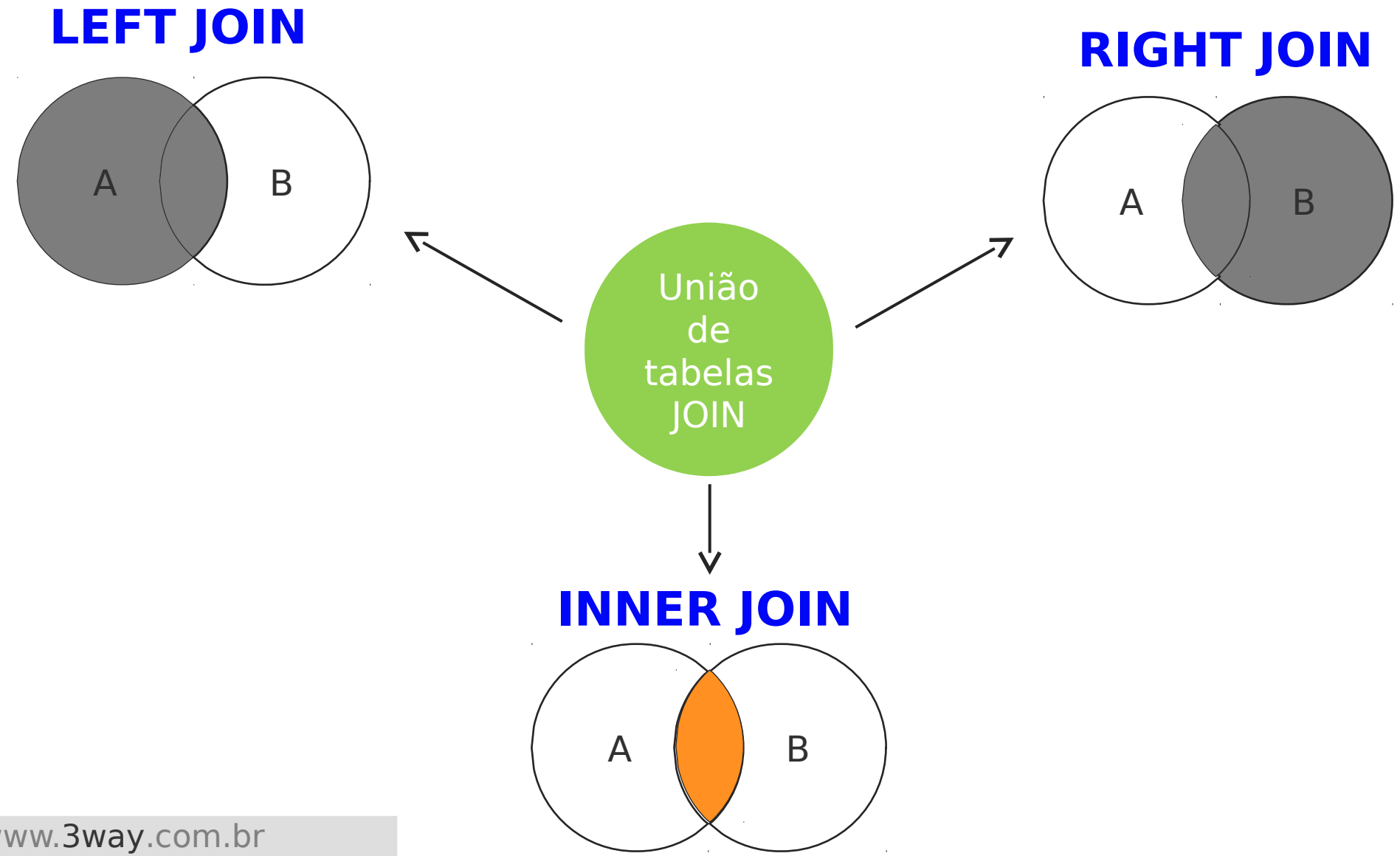
Coluna "id"

Tabela-clientes			
ID	NOME	ENDEREÇO	CPF
001	João	XXX	123456
002	**Maria**	**YYY**	**654321**
003	Paulo	AAA	000000

Resultado do DELETE

Tabela-clientes			
ID	NOME	ENDEREÇO	CPF
001	João	XXX	123456
003	Paulo	AAA	000000

União de tabelas JOIN



União de tabelas – INNER JOIN

```
SELECT * FROM clientes INNER JOIN enderecos ON clientes.fkEndereco = enderecos.idEndereco
```

Tabela- Clientes

ID	NOME	FKEndereco	CPF
001	João	001	123456
003	Paulo	002	000000

Chave estrangeira

Tabela-endereco

IDEndereco	BAIRRO	CIDADE
001	X	XX
002	A	AA

Chave primária

Resultado do INNER JOIN

ID	NOME	FKEndereço	CPF	ID endereco	BAIRRO	CIDADE
001	João	001	123456	001	X	XX
003	Paulo	002	000000	002	A	AA

União de tabelas – RIGHT JOIN

```
SELECT * FROM clientes RIGHT JOIN enderecos ON clientes.fkEndereco = enderecos.idEndereco
```

Tabela- Clientes			
ID	NOME	FKEndereco	CPF
001	João	001	123456
003	Paulo	002	000000
004	Maria	004	456789

Chave estrangeira

Tabela-endereco		
IDEndereço	BAIRRO	CIDADE
001	X	XX
002	A	AA
003	Y	YY

Chave primária

Resultado do RIGHT JOIN

ID	NOME	FKEndereco	CPF	ID endereco	BAIRRO	CIDADE
001	João	001	123456	001	X	XX
003	Paulo	002	000000	002	A	AA
				003	Y	YY

União de tabelas – LEFT JOIN

```
SELECT * FROM clientes LEFT JOIN enderecos ON clientes.fkEndereco = enderecos.idEndereco
```

Tabela- Clientes			
ID	NOME	FKEndereço	CPF
001	João	001	123456
003	Paulo	002	000000
004	Maria	004	456789

Tabela-endereco		
IDEndereço	BAIRRO	CIDADE
001	X	XX
002	A	AA
003	Y	YY

Chave estrangeira

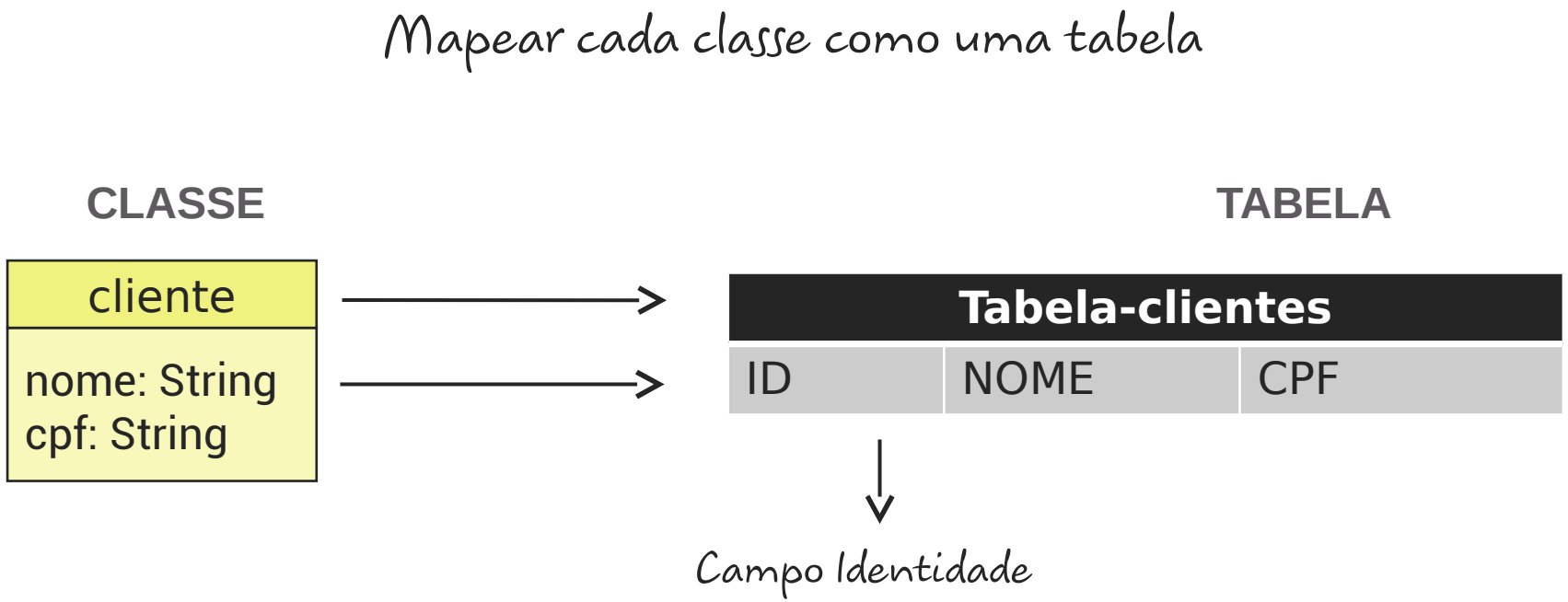
Chave primária

Resultado do LEFT JOIN

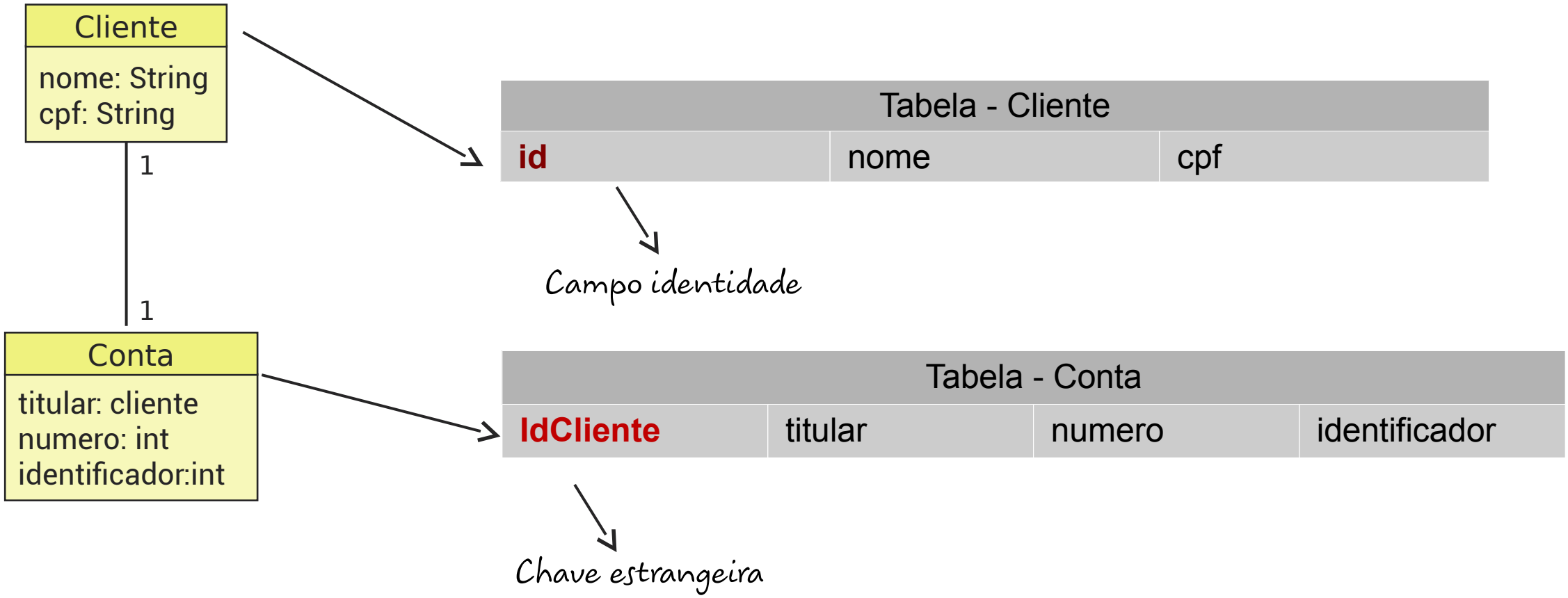
ID	NOME	FKEndereço	CPF	ID endereco	BAIRRO	CIDADE
001	João	001	123456	001	X	XX
003	Paulo	002	000000	002	A	AA
004	Maria	004	456789			

Objeto Relacional

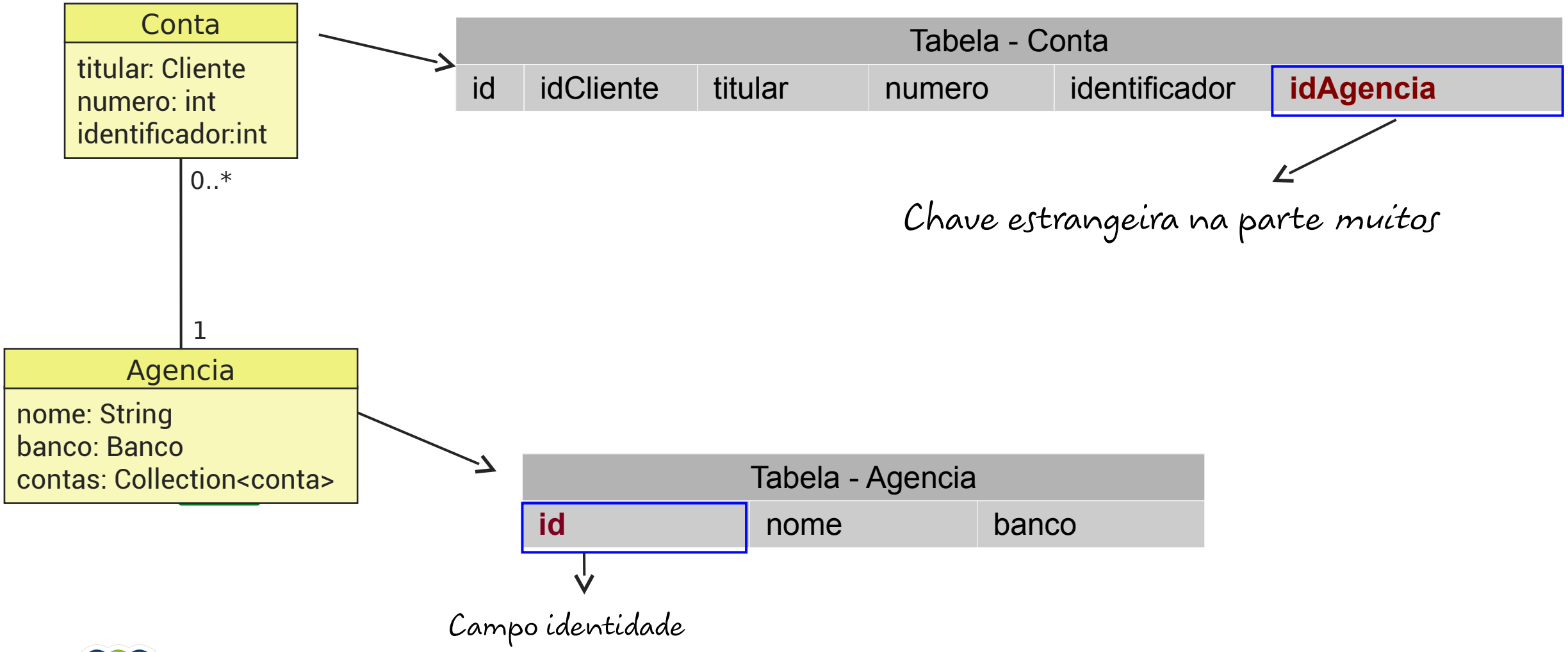
Mapeamento das classes e seus tributos



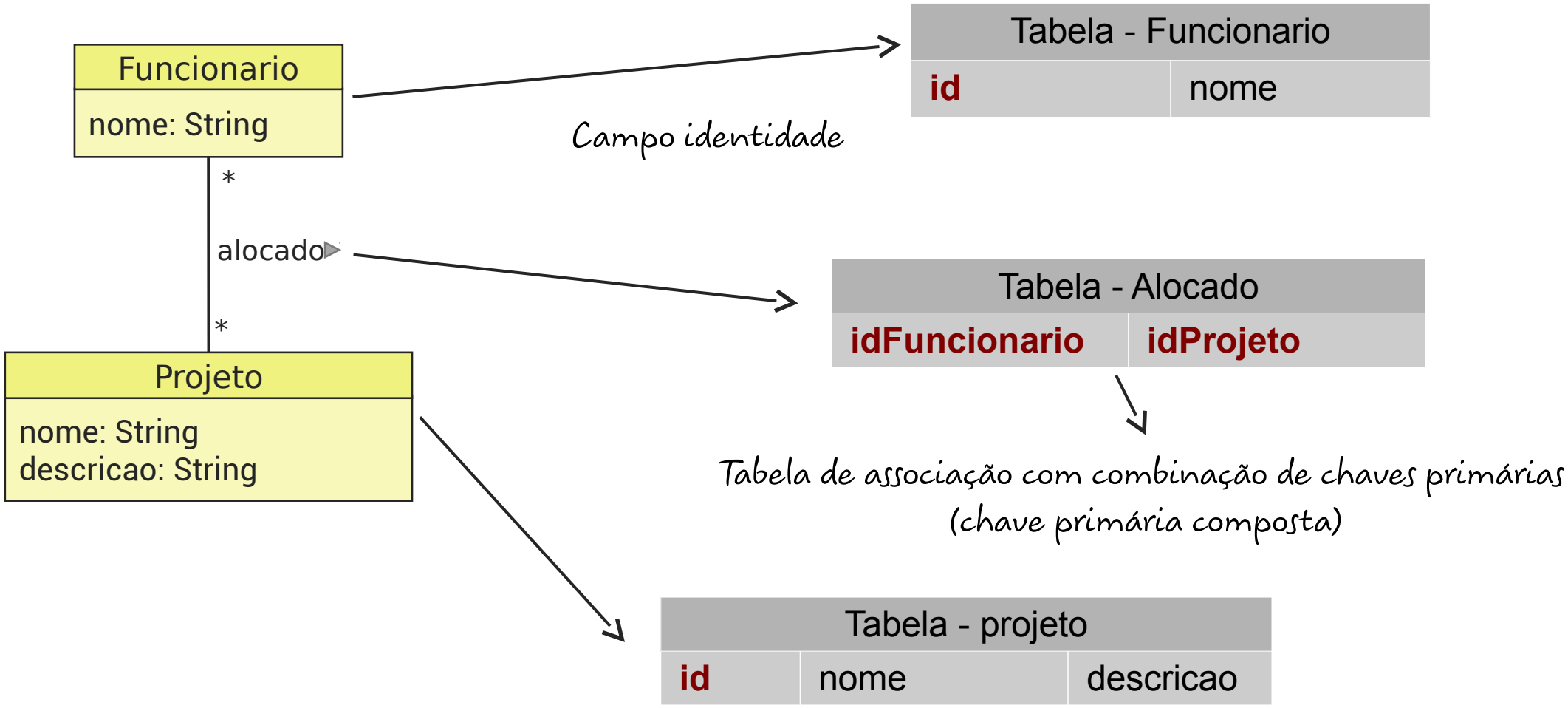
Mapeamento de Associação 1-1



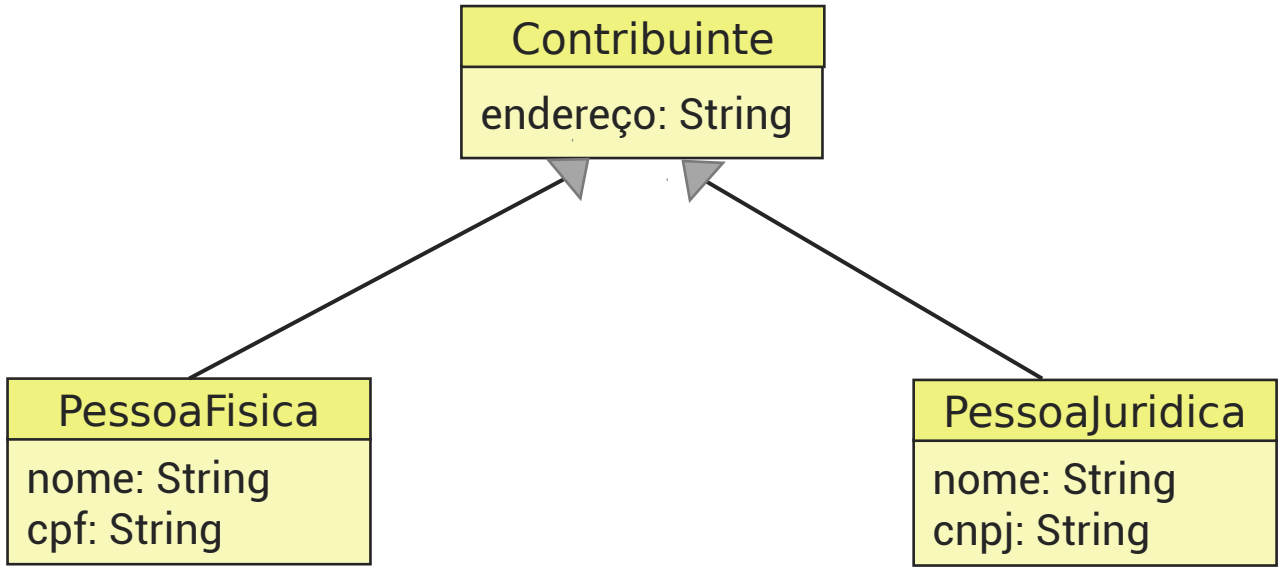
Mapeamento de Associação 1 - muitos



Mapeamento de Associação muitos - muitos

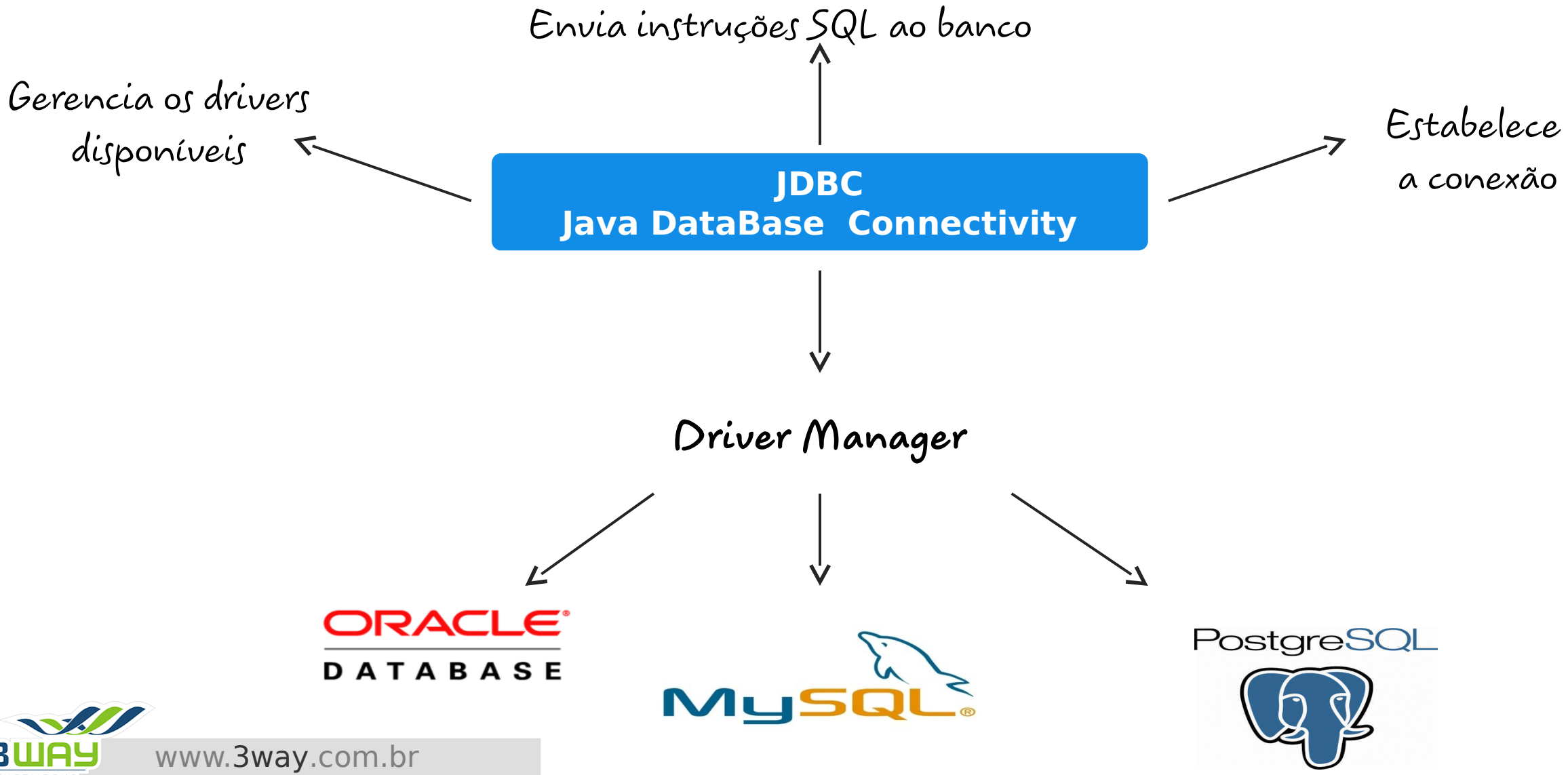


Mapeamento de Herança

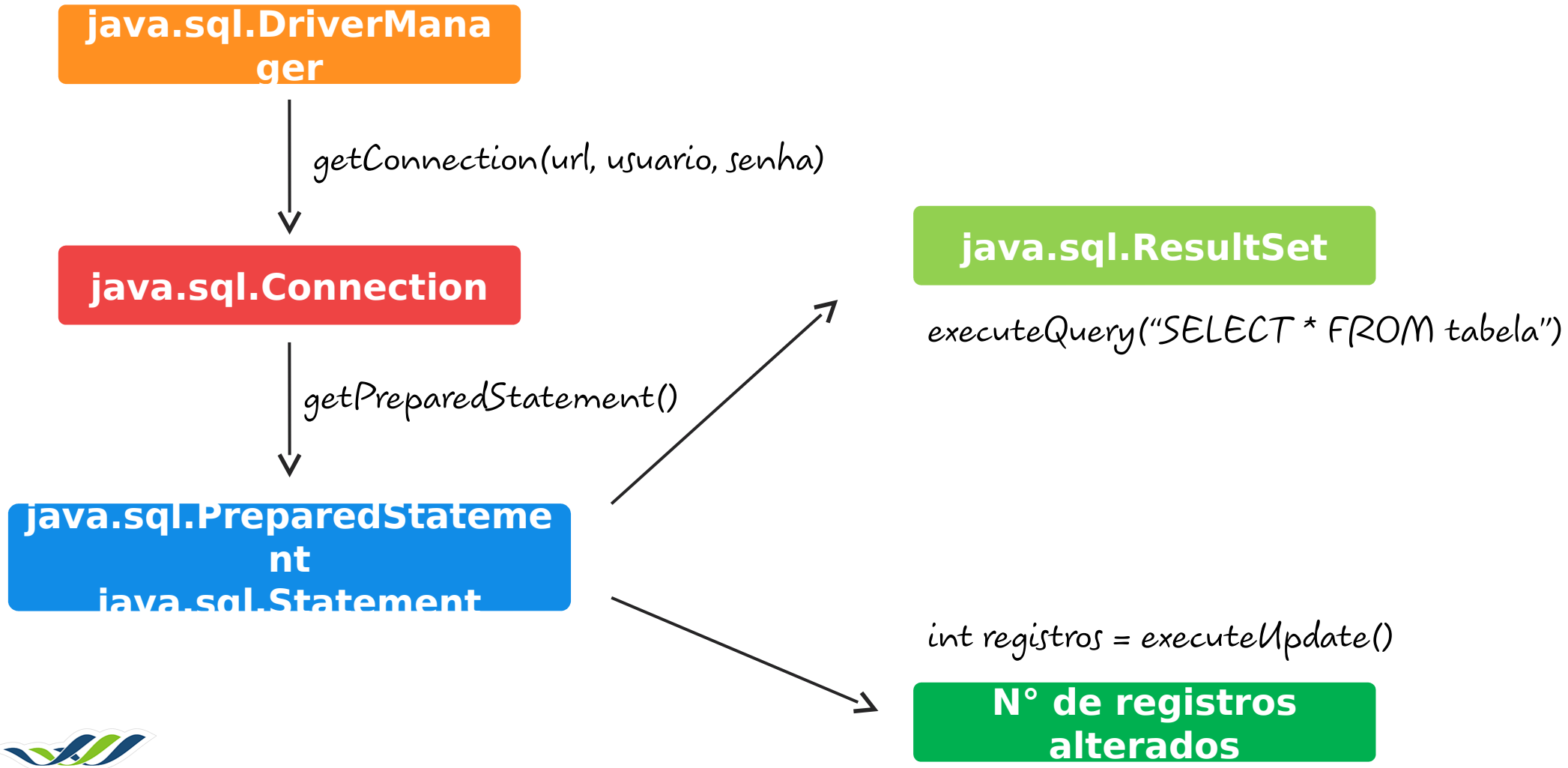


3 Possíveis Soluções

JDBC Java DataBase Connectivity



Prepared Statement



Padrão Fábrica – Conexão Com BD

```java
package threeway.projeto.service.Dao;

import java.sql.Connection;

public class FabricaConexao {

    public static String url = "jdbc:postgresql://localhost:5432/threewayweb";
    public static String usuario = "postgres";
    public static String senha = "123456";

    public static Connection getConexao() throws SQLException {

        try{

            Class.forName("org.postgresql.Driver");

            return DriverManager.getConnection(url,usuario,senha);

        }catch(ClassNotFoundException e){
            throw new SQLException(e.getMessage());
        }
    }
}
```

Dados da conexão:

- Nome da base de dados
- Usuário
- Senha

Retorna a conexão usando o DriverManager

SQL no Código

Pegue a conexão antes de fazer o Prepared Statement

Cria o obejto PreparedStatement com a SQL parametrizada

executeQuery() executa a SQL presente no objeto PreparedStatement e retorna um objeto ResultSet como resultado

SQL

```
private static final String SELECT_SQL = "SELECT NOME, SENHA, LOGIN, "
        + "ENDERECO, CIDADE, BAIRRO, ESTADO, "
        + "CEP, COD_CLIENTE FROM CLIENTE WHERE SENHA = ? and LOGIN = ? ";

public Cliente obter(Cliente cliente) {

    try (Connection conexao = FabricaConexao.getConexao();
        PreparedStatement consulta = conexao.prepareStatement(SELECT_SQL)) {

        consulta.setString(1, cliente.getSenha());
        consulta.setString(2, cliente.getLogin());
        ResultSet resultado = consulta.executeQuery();

        if (resultado.next()) {
            cliente.setNome(resultado.getString("NOME"));
            cliente.setSenha(resultado.getString("SENHA"));
            cliente.setLogin(resultado.getString("LOGIN"));
            cliente.setEndereco(resultado.getString("ENDERECO"));
            cliente.setCidade(resultado.getString("CIDADE"));
            cliente.setBairro(resultado.getString("BAIRRO"));
            cliente.setEstado(resultado.getString("ESTADO"));
            cliente.setCep(resultado.getString("CEP"));
            cliente.setCodigo(resultado.getInt("COD_CLIENTE"));
            cliente.setAutenticacao(true);
        } else {
            cliente = null;
        }
    } catch (SQLException e) {
        log.severe(e.getMessage());
    }
    return cliente;
}
```

Prepared Statement para Usando JDBC

```
private static final String INSERT_SQL = "INSERT INTO CLIENTE "
        + "(NOME,LOGIN,SENHA,ENDERECO,CIDADE,BAIRRO,ESTADO,CEP) "
        + "VALUES (?,?,?,?,?,?,?,?)";
```

SQL

Pegue a conexão antes de fazer o Prepared Statement

Cria o obejto PreparedStatement com a SQL parametrizada

ExecuteUpdate() executa a SQL presente no objeto PreparedStatement

```
public Cliente inserir(Cliente cliente) {
    try (Connection conexao = FabricaConexao.getConexao();
         PreparedStatement consulta = conexao.prepareStatement(INSERT_SQL)
        ) {

        consulta.setString(1, cliente.getNome());
        consulta.setString(2, cliente.getLogin());
        consulta.setString(3, cliente.getSenha());
        consulta.setString(4, cliente.getEndereco());
        consulta.setString(5, cliente.getCidade());
        consulta.setString(6, cliente.getBairro());
        consulta.setString(7, cliente.getEstado());
        consulta.setString(8, cliente.getCep());

        consulta.executeUpdate();

    } catch (SQLException e) {
        log.severe(e.getMessage());
    }
    return cliente;
}
```

SQL no Código

```
private static final String UPDATE_SQL= "UPDATE cliente SET "+
        "SENHA = ?, LOGIN = ?, ENDERECO = ?, CIDADE = ?, "+
        "BAIRRO = ?, ESTADO = ?, CEP = ? "+
        "WHERE COD_CLIENTE = ?";
```

SQL

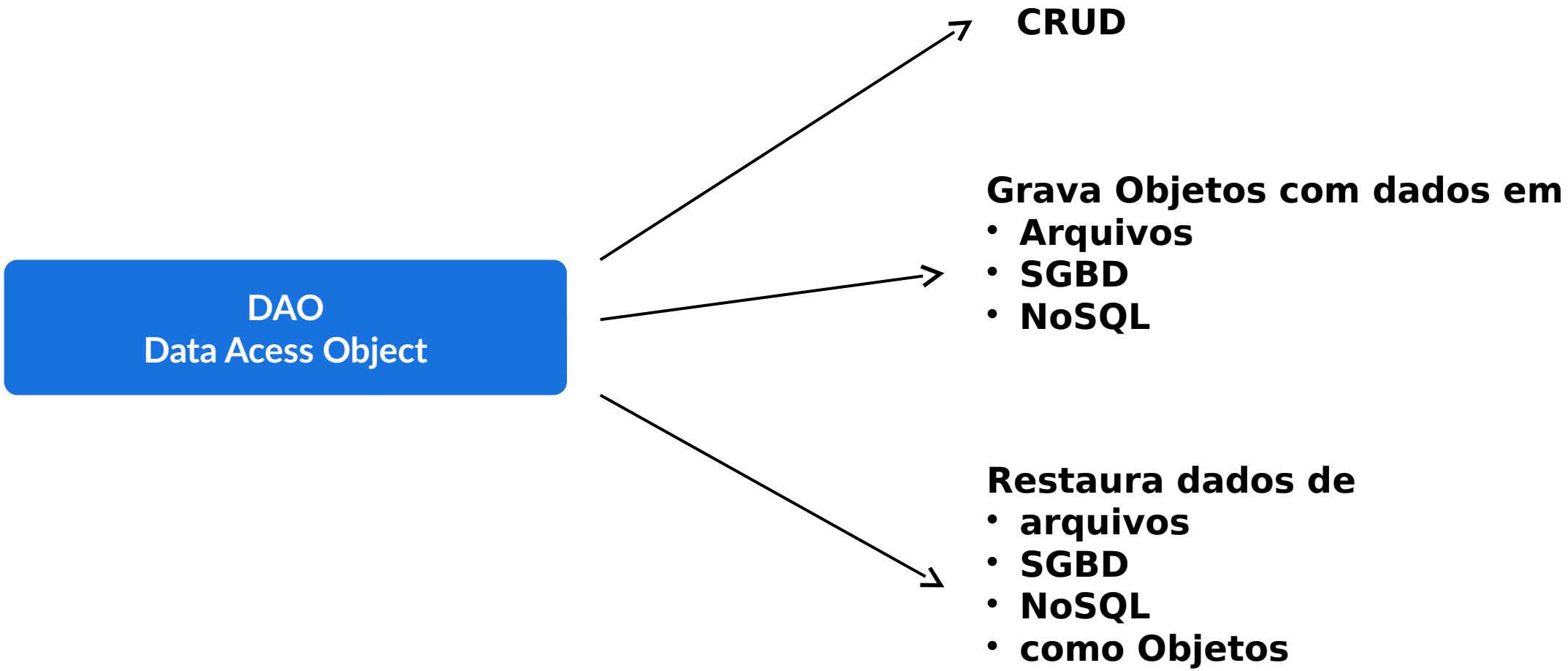
Pegue a conexão antes de fazer o Prepared Statement

Cria o obejto PreparedStatement com a SQL parametrizada

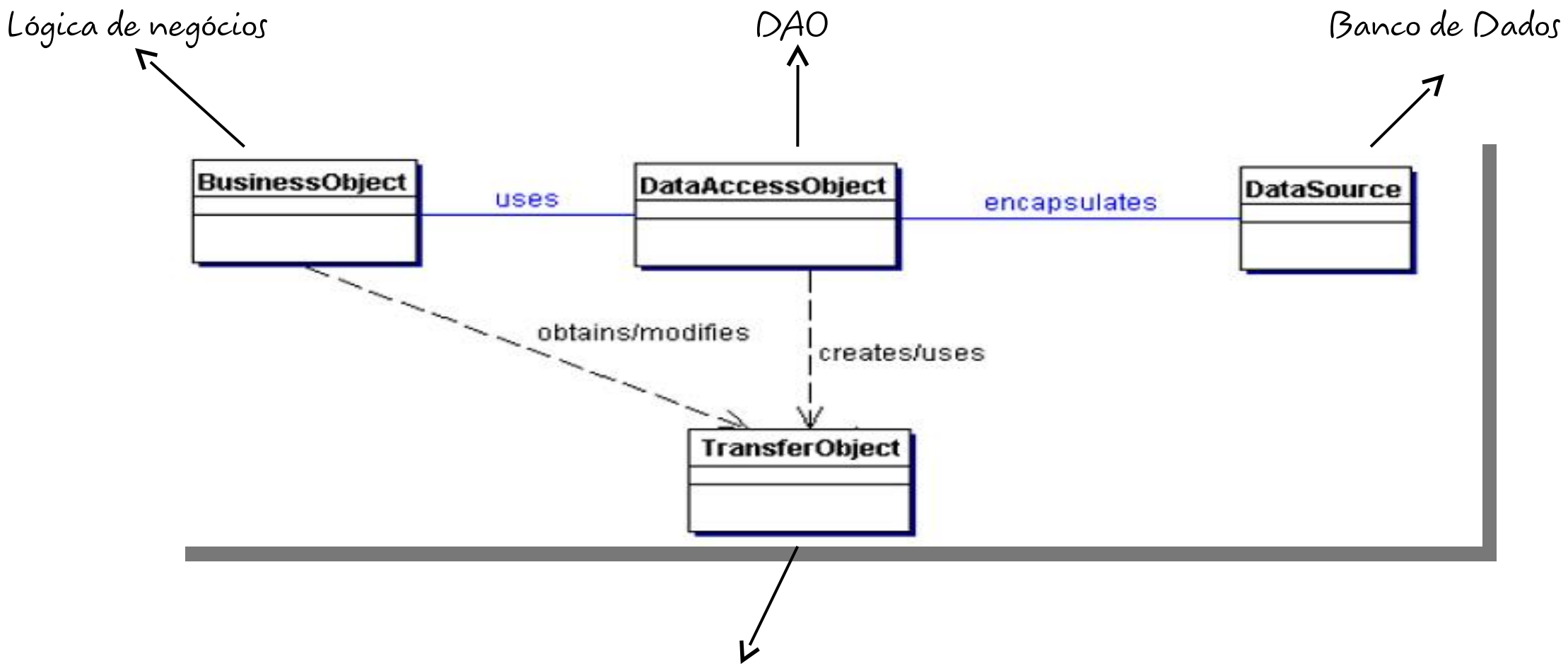
ExecuteUpdate() executa a SQL presente no objeto PreparedStatement

```
public void alterar(Cliente cliente) {

    try (Connection conexao = FabricaConexao.getConexao();
        PreparedStatement consulta = conexao.prepareStatement(UPDATE_SQL)

        ){

        consulta.setString(1, cliente.getSenha());
        consulta.setString(2, cliente.getLogin());
        consulta.setString(3, cliente.getEndereco());
        consulta.setString(4, cliente.getCidade());
        consulta.setString(5, cliente.getBairro());
        consulta.setString(6, cliente.getEstado());
        consulta.setString(7, cliente.getCep());
        consulta.setInt(8, cliente.getCodigo());

        consulta.execute();

    } catch (SQLException e) {
        log.severe(e.getMessage());
    }
}
```

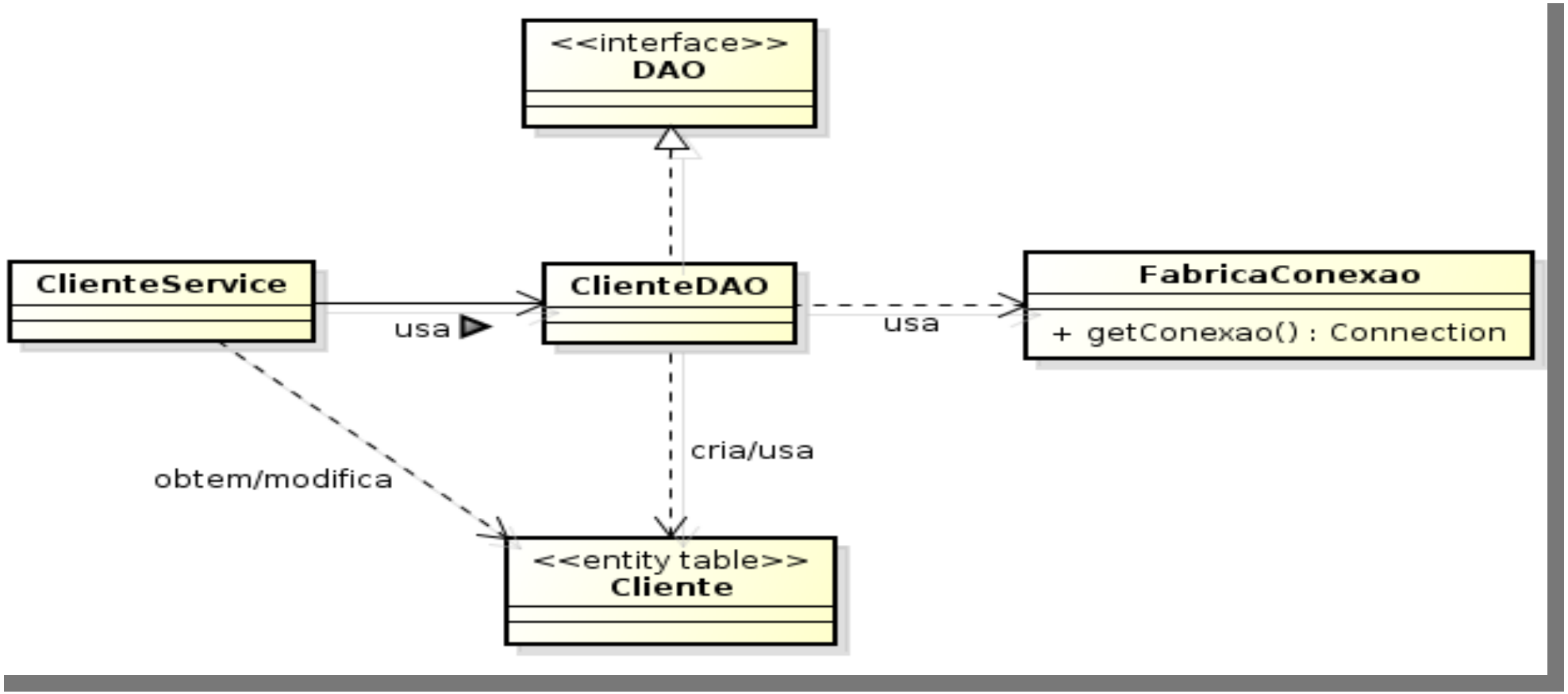
DAO Data Acess Object



Padrão DAO


Lógica de negócios
DAO
Banco de Dados
BusinessObject
uses
DataAccessObject
encapsulates
DataSource
obtains/modifies
creates/uses
TransferObject
Passa e recebe dados do cliente

Padrão DAO


<<interface>>
DAO
ClienteService
ClienteDAO
FabricaConexao
+ getConexao() : Connection
usa
usa
obtem/modifica
cria/usa
<<entity table>>
Cliente

CRUD CRIAR

```
public void salvar(Cliente entidade) {
    StringBuffer sql = new StringBuffer();
    sql.append("INSERT INTO cliente");
    sql.append("(endereco, nome, telefone, cpf, rg) VALUES (?,?,?,?,?)");
    try {
        PreparedStatement consulta = conexao.prepareStatement(sql.toString());
        consulta.setString(1, entidade.getEndereco());
        consulta.setString(2, entidade.getNome());
        consulta.setString(3, entidade.getTelefone());
        consulta.setString(4, entidade.getCpf());
        consulta.setString(5, entidade.getRg());
        consulta.executeUpdate();
    } catch (SQLException e) {
        System.out.println("Erro ao realizar Insert: " + e.getMessage());
    }
}
```

CRUD RECUPERAR

```
public Cliente obter(Serializable identificador) {
    Cliente cliente = null;
    String sql = "SELECT * FROM cliente WHERE identificador = ?";
    try {
        PreparedStatement consulta = conexao.prepareStatement(sql);
        consulta.setLong(1, (Long) identificador);
        ResultSet resultado = consulta.executeQuery();
        if(resultado.next()){
            cliente = new Cliente();
            cliente.setIdentificador(resultado.getLong("identificador"));
            cliente.setEndereco(resultado.getString("endereco"));
            cliente.setNome(resultado.getString("nome"));
            cliente.setTelefone(resultado.getString("telefone"));
            cliente.setCpf(resultado.getString("cpf"));
            cliente.setRg(resultado.getString("rg"));
        }
    } catch (SQLException e) {
        System.out.println("Erro ao realizar Select: " + e.getMessage());
    }
    return cliente;
}
```

CRUD UPDATE

```java
public void alterar(Cliente entidade) {

    StringBuilder sql = new StringBuilder();

    sql.append("UPDATE cliente SET ");
    sql.append("endereco = ?, ");
    sql.append("nome = ?, ");
    sql.append("telefone = ?, ");
    sql.append("cpf = ?, ");
    sql.append("rg = ? ");
    sql.append("WHERE identificador = ?");

    try {

        PreparedStatement consulta = conexao.prepareStatement(sql.toString());

        consulta.setString(1, entidade.getEndereco());

        consulta.setString(2, entidade.getNome());

        consulta.setString(3, entidade.getTelefone());

        consulta.setString(4, entidade.getCpf());

        consulta.setString(5, entidade.getRg());

        consulta.setLong(6, entidade.getIdentificador());

        consulta.executeUpdate();


    } catch (SQLException e) {

        System.out.println("Erro ao realizar Update: " + e.getMessage());
    }

}
```

CRUD DELETAR

```
public void remover(Cliente entidade) {
    String sql = "DELETE FROM cliente WHERE identificador = ?";
    try {
        PreparedStatement consulta = conexao.prepareStatement(sql);
        consulta.setLong(1, entidade.getIdentificador());
        consulta.executeUpdate();
    } catch (SQLException e) {
        System.out.println("Erro ao realizar Delete: " + e.getMessage());
    }
}
```