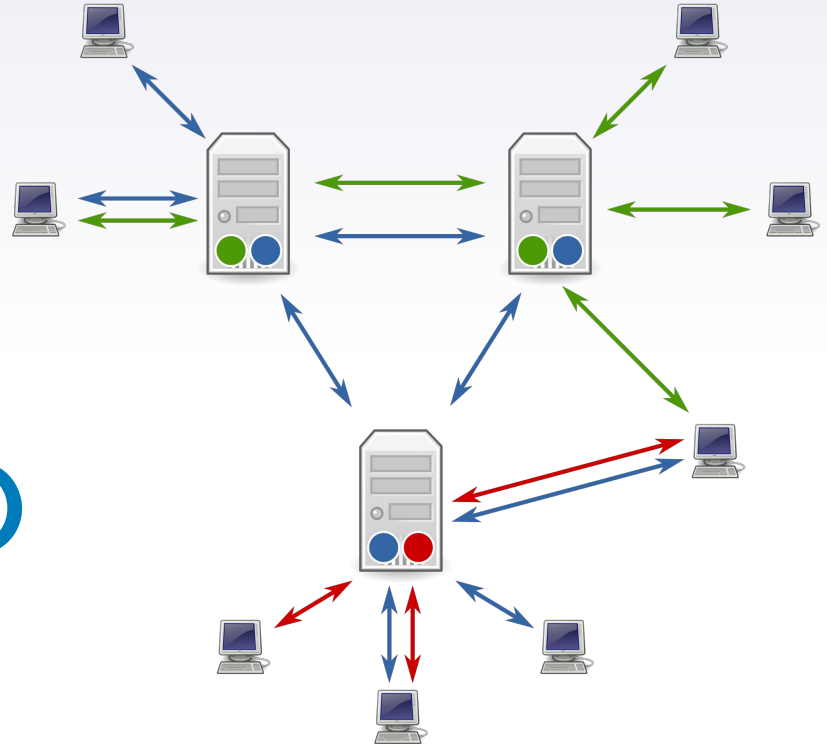


SISTEMA BANCÁRIO DISTRIBUÍDO



▶ Alunos

- ▶ Isabela Fontes de Araújo
- ▶ Kéthlyn Campos Silva
- ▶ Kamilla Cristina Silva Martins
- ▶ Fabrício Lopes Mendonça
- ▶ João Victor Santos Alves e Silva



Introdução

- ▶ O projeto desenvolvido é um Sistema Bancário Distribuído (SBD);
- ▶ Requisitos do Sistema:
 - ▷ Abertura de conta;
 - ▷ Consulta de saldo e extrato da conta
 - ▷ Transferências entre contas
 - ▷ Saque
 - ▷ Depósito



Metodologia de Trabalho

- ▶ Java;
- ▶ Java Remote Method Invocation (RMI);
- ▶ Java Database Connectivity (JDBC).

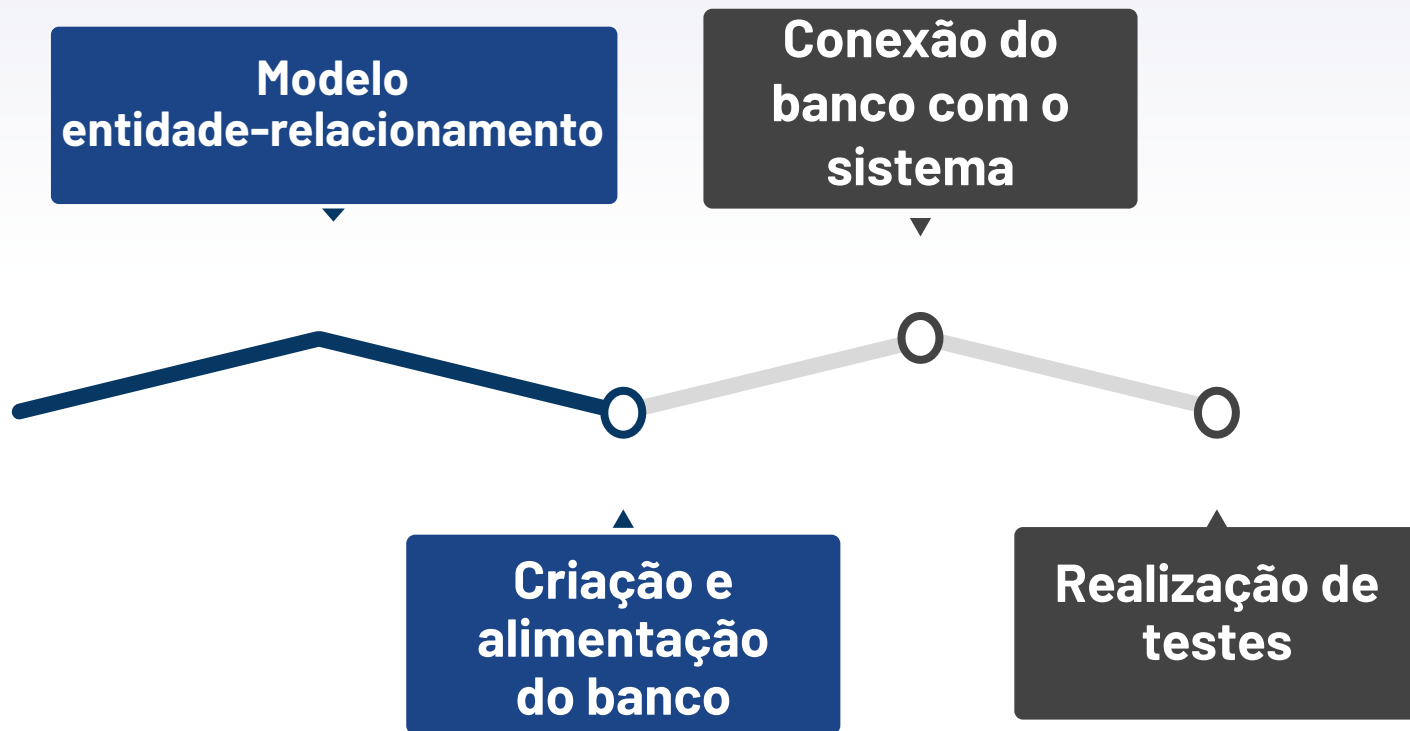


Vantagens

- ▶ Sistema Distribuído
 - ▷ Disponibilidade de serviços
 - ▷ Heterogeneidade
 - ▷ Transparência
 - ▷ Independência
- ▶ Java RMI
 - ▷ Atende aos requisitos
 - ▷ Comunicação distribuída

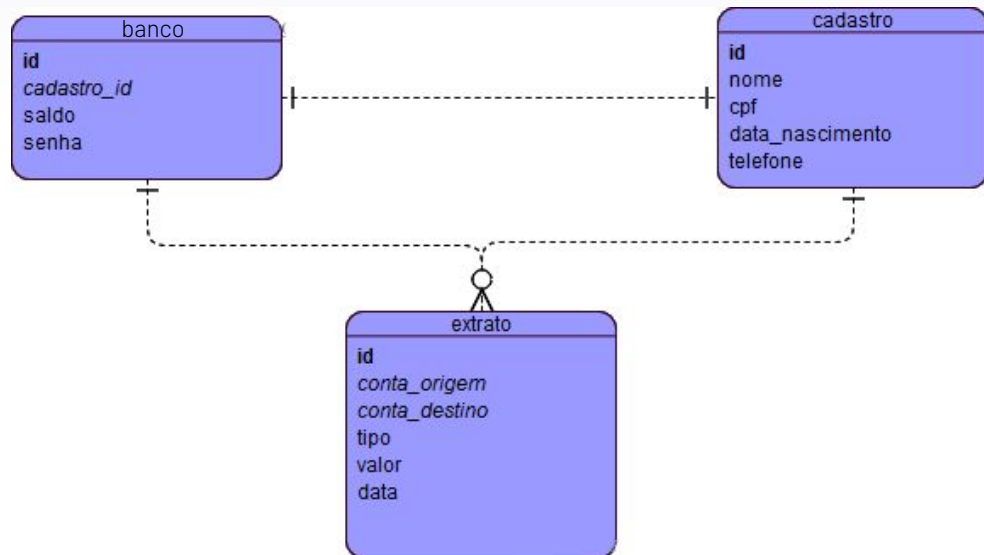


► Divisão de etapas



Arquitetura do Software

- ▶ Modelo entidade-relacionamento



▶ Código SQL

- ▶ Criar tabelas:

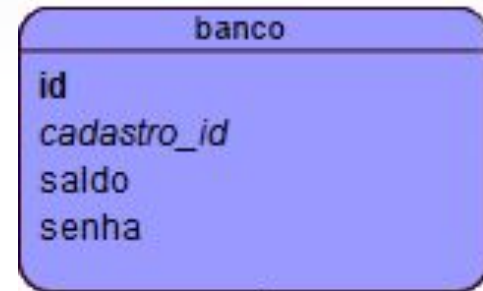
```
CREATE TABLE cadastro(  
    id SERIAL PRIMARY KEY,  
    nome VARCHAR(50) NOT NULL,  
    cpf VARCHAR(11) UNIQUE NOT NULL,  
    telefone VARCHAR(13) NOT NULL,  
    data_nascimento DATE NOT NULL  
);
```



▶ Código SQL

- ▶ Criar tabelas:

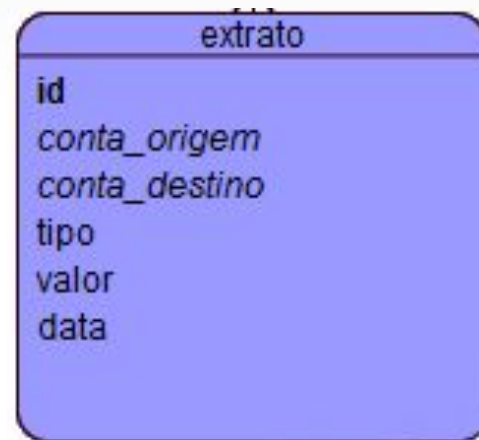
```
CREATE TABLE banco(  
    id SERIAL PRIMARY KEY,  
    cadastro_id INT REFERENCES cadastro(id),  
    saldo numeric(15,2) default 0,  
    senha VARCHAR NOT NULL  
);
```



▶ Código SQL

- ▶ Criar tabelas:

```
CREATE TABLE extrato(  
    id SERIAL PRIMARY KEY,  
    conta_origem INT REFERENCES cadastro(id),  
    conta_destino INT REFERENCES cadastro(id),  
    tipo VARCHAR NOT NULL,  
    valor numeric(15,2) default 0,  
    data DATE NOT NULL  
);
```

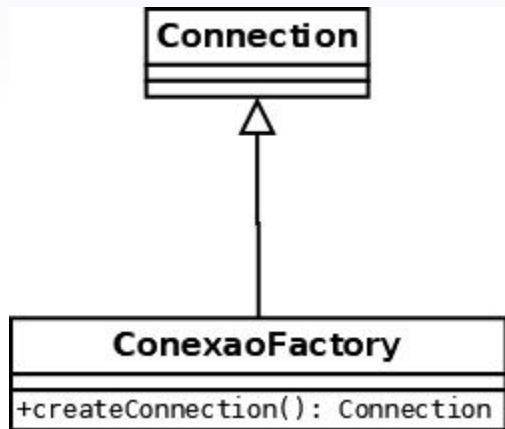


Diagramas de Classes

- ▶ ConexaoFactory;
- ▶ AbstractRepository;
- ▶ ExtratoRepository;
- ▶ CadastroRepository;
 - ▶ SqlBuilder.
- ▶ BancoRepository.
 - ▶ Banco.
 - ▶ BancoController.

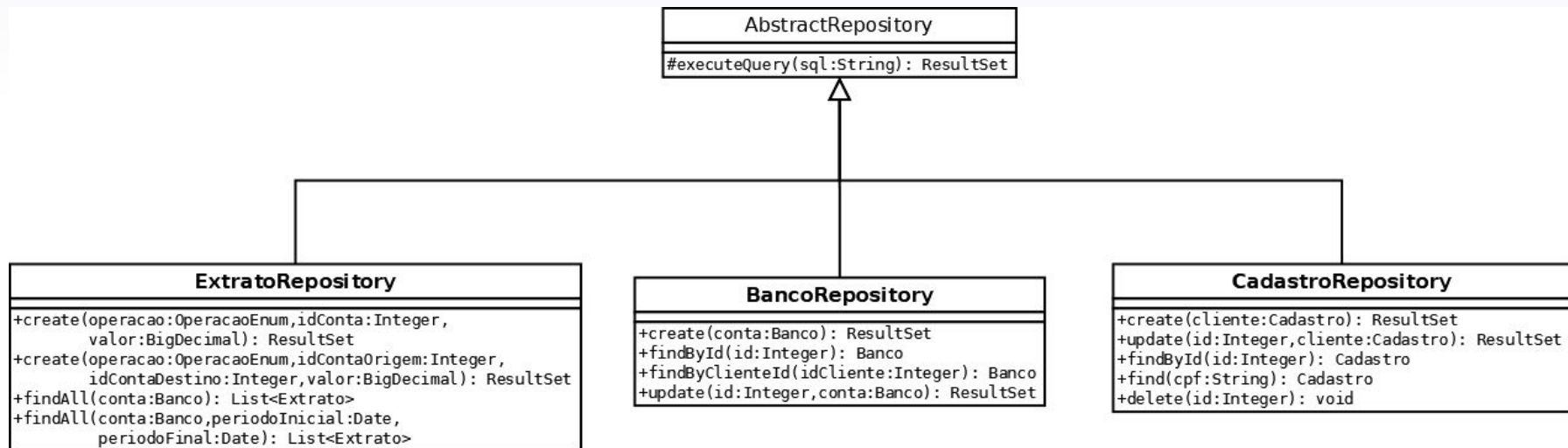


► ConexaoFactory

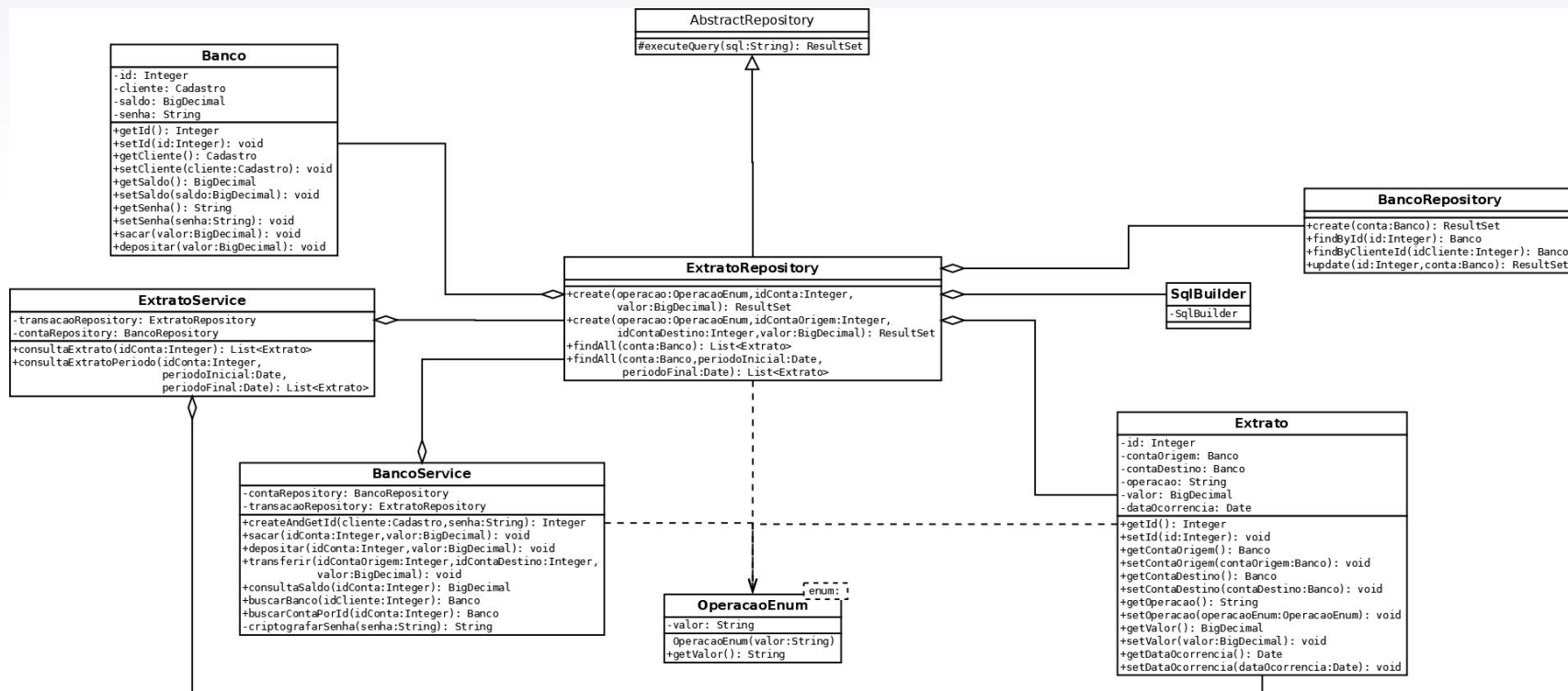


```
1 package br.com.banco.server;
2
3 import java.sql.*;
4
5 public class ConexaoFactory {
6
7     public static Connection createConnection() throws SQLException{
8         String url = "jdbc:postgresql://172.16.56.190/postgres";
9         String user = "postgres";
10        String password = "123456";
11
12        return DriverManager.getConnection(url, user, password);
13    }
14 }
```

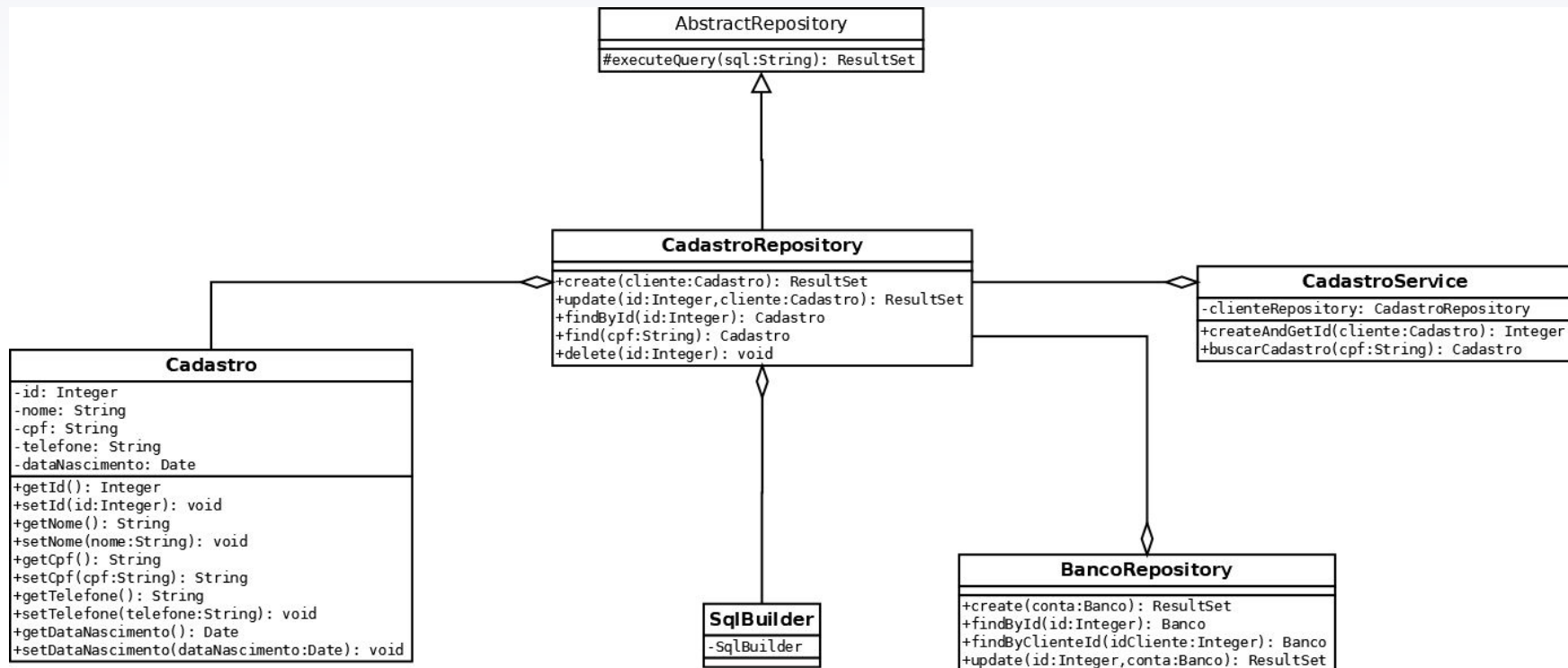
► AbstractRepository



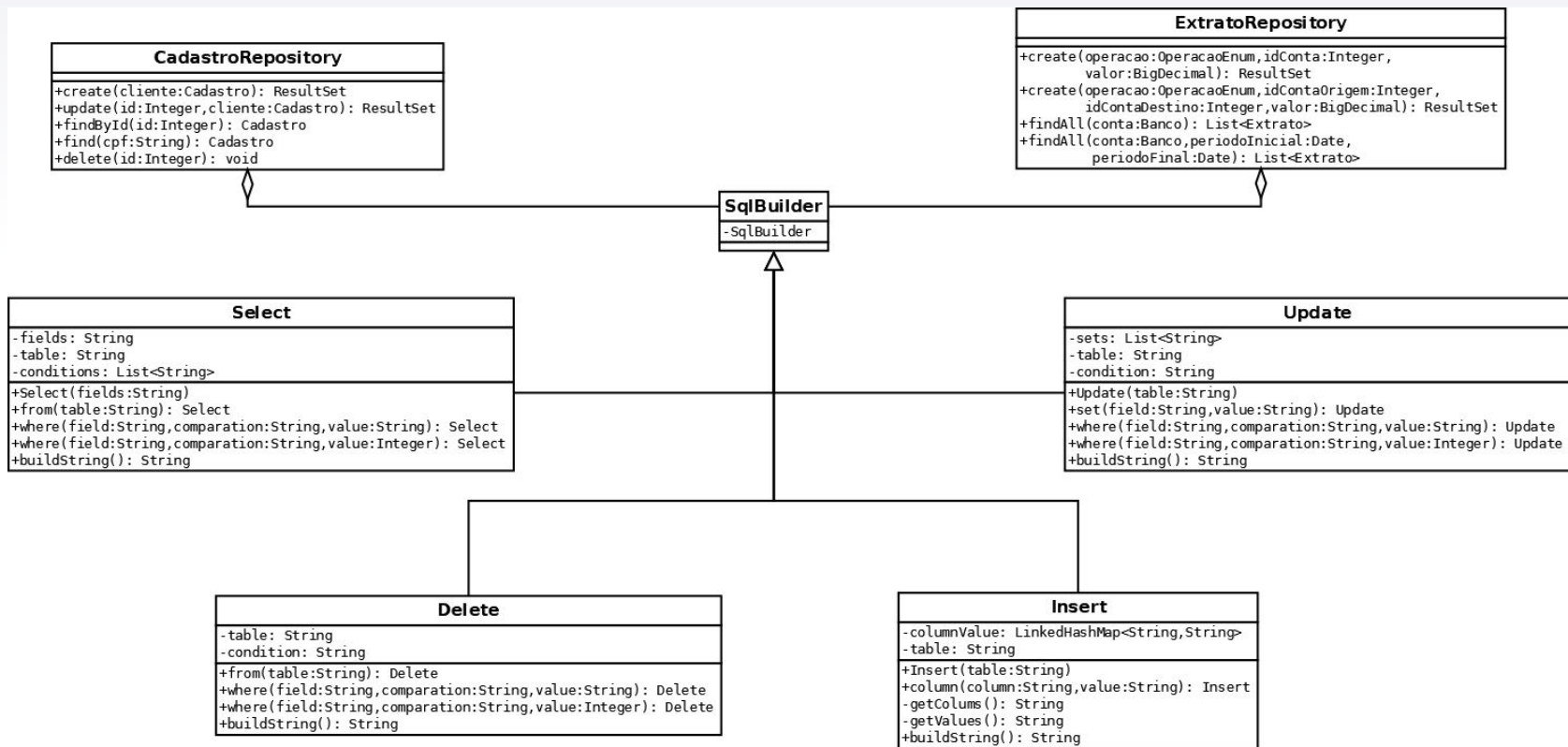
ExtratoRepository



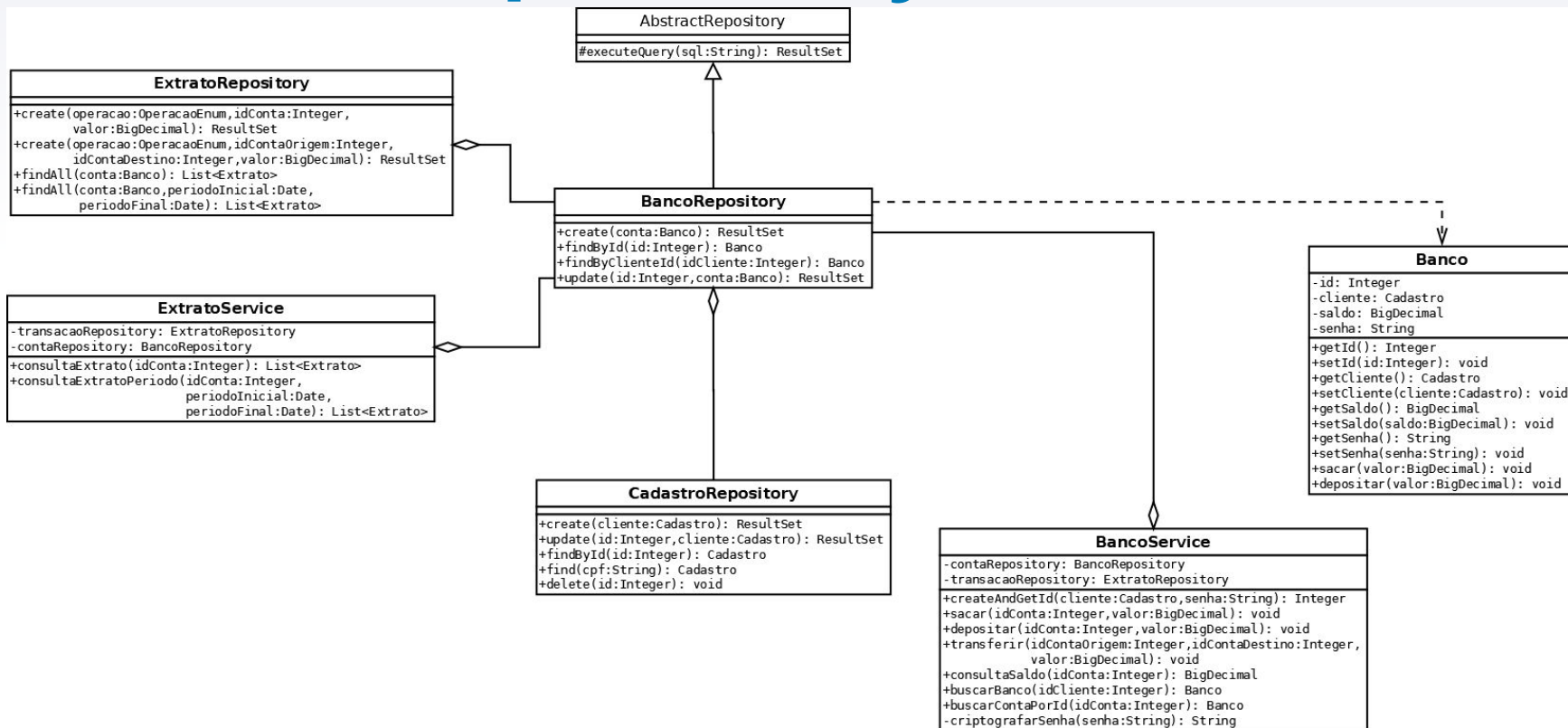
CadastroRepository



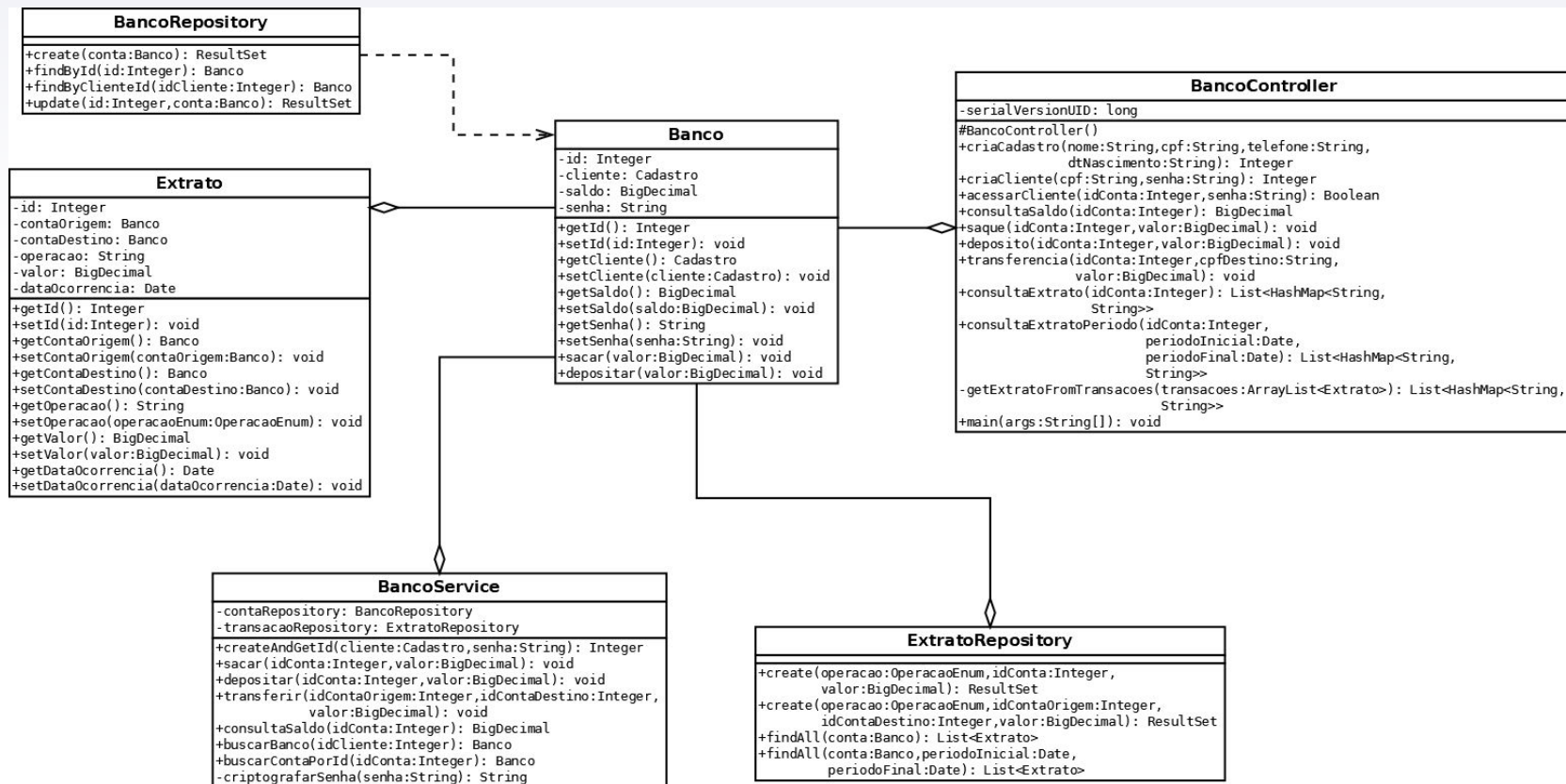
SqlBuilder



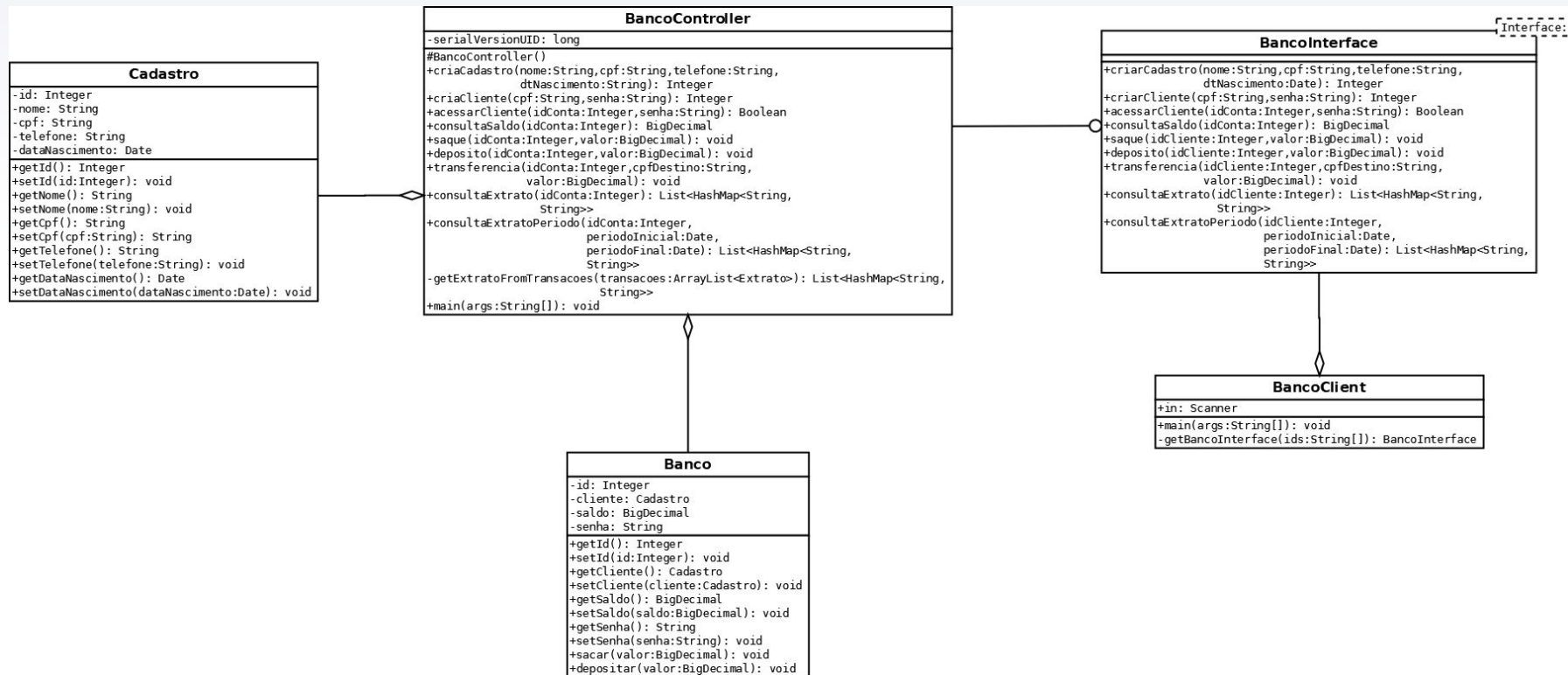
BancoRepository



Banco

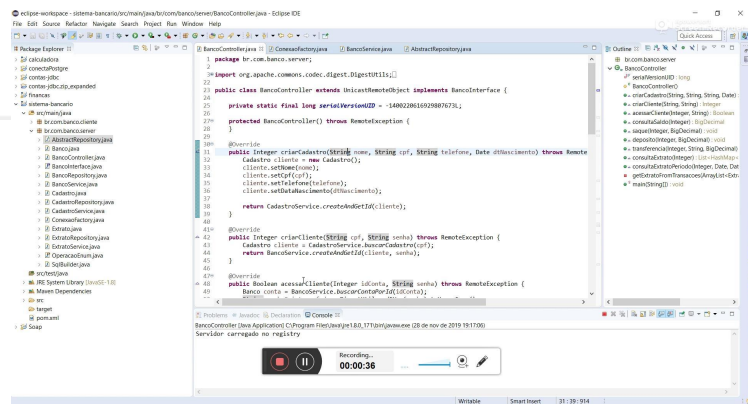


BancoController



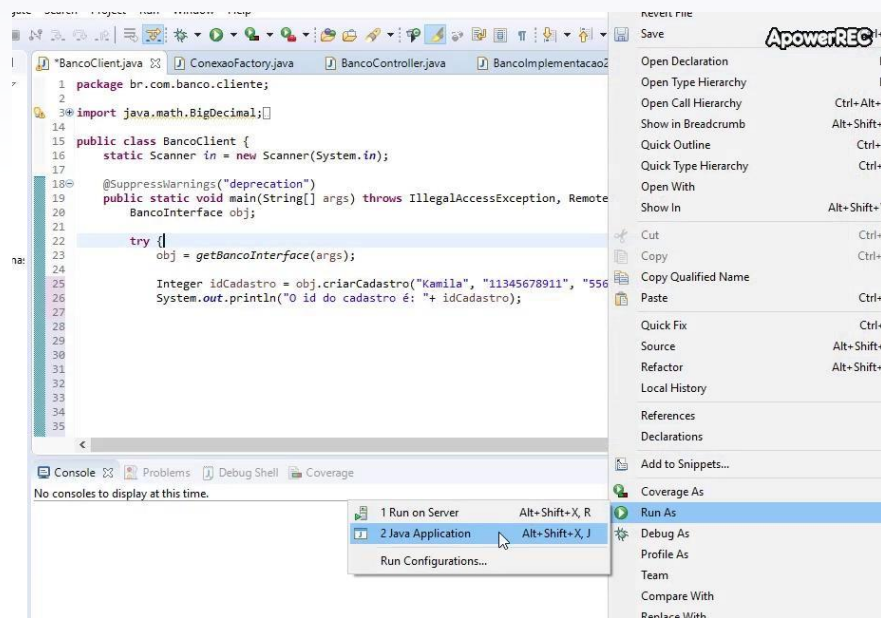
BancoController

```
107- public static void main(String[] args) {  
108     try {  
109         BancoController obj = new BancoController();  
110         Registry registry = LocateRegistry.createRegistry(2001);  
111         registry.rebind("BancoServer", obj);  
112         System.out.println("Servidor carregado no registry");  
113     } catch (Exception e) {  
114         System.out.println("Banco RMI erro: " + e.getMessage());  
115     }  
116 }  
117 }  
118 }
```



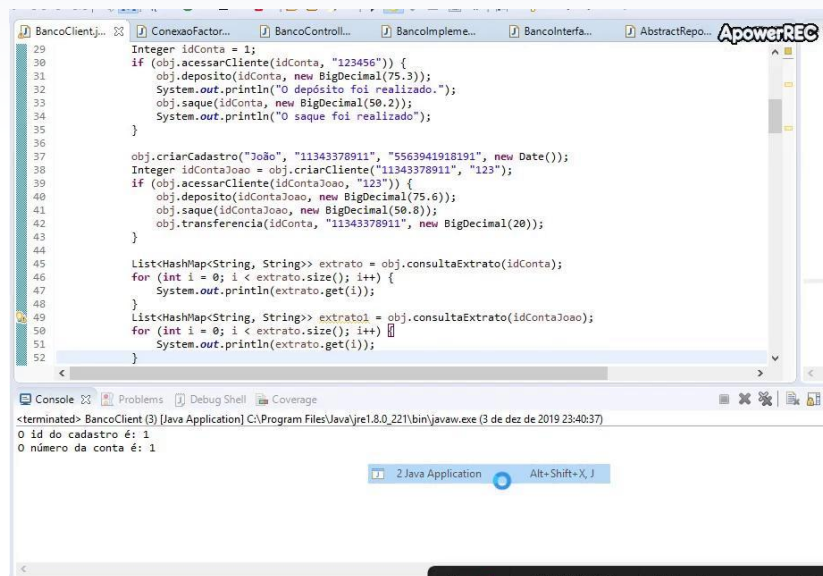
Alguns testes

Criação de contas



Alguns testes

Depósito e Saque



The screenshot shows an IDE with a Java file named `BancoClient.java`. The code includes methods for depositing and withdrawing money, as well as creating a client and checking account balances. The console output shows the results of running the application, including the creation of a client and the execution of deposit and withdrawal operations.

```
29 Integer idConta = 1;
30 if (obj.acessarCliente(idConta, "123456")) {
31     obj.deposito(idConta, new BigDecimal(75.3));
32     System.out.println("O depósito foi realizado.");
33     obj.saque(idConta, new BigDecimal(50.2));
34     System.out.println("O saque foi realizado");
35 }
36
37 obj.criarCadastro("João", "11343378911", "5563941918191", new Date());
38 Integer idContaJoao = obj.criarCliente("11343378911", "123");
39 if (obj.acessarCliente(idContaJoao, "123")) {
40     obj.deposito(idContaJoao, new BigDecimal(75.6));
41     obj.saque(idContaJoao, new BigDecimal(50.8));
42     obj.transferencia(idConta, "11343378911", new BigDecimal(20));
43 }
44
45 List<HashMap<String, String>> extrato = obj.consultaExtrato(idConta);
46 for (int i = 0; i < extrato.size(); i++) {
47     System.out.println(extrato.get(i));
48 }
49 List<HashMap<String, String>> extrato1 = obj.consultaExtrato(idContaJoao);
50 for (int i = 0; i < extrato1.size(); i++) {
51     System.out.println(extrato1.get(i));
52 }
```

Console Output:

```
<terminated> BancoClient (3) [Java Application] C:\Program Files\Java\jre1.8.0_221\bin\javaw.exe (3 de dez de 2019 23:40:37)
O id do cadastro é: 1
O número da conta é: 1
```

Conclusão

- ▶ Sistema distribuído lógica e fisicamente ✓
- ▶ Requisitos do sistema:
 - ▷ Abertura de conta ✓
 - ▷ Consulta de saldo e extrato da conta ✓
 - ▷ Transferências entre contas ✓
 - ▷ Saque ✓
 - ▷ Depósito ✓

Obrigado pela
atenção!