

Universidade Presbiteriana Mackenzie
Faculdade de Computação e Informática

Aplicações Java com persistência em base de dados

NOTAS DE AULA - Teoria 07

Linguagem de Programação II
2º semestre de 2015
Prof. Tomaz Mikio Sasaki
Versão 1.0

Objetivos

- Criar tabelas em base de dados para persistir informações da aplicação.
- Executar sentenças SQL básicas.
- Desenvolver aplicações Java com persistência em base de dados.

Referência

A referência para esta aula é o capítulo 4 de:

HORSTMANN, C.S.; CORNELL, G. **Core Java, Volume II - Advanced Features** Prentice Hall, 2013.

Observação: As notas de aula são material de apoio para estudo e não têm o objetivo de apresentar o assunto de maneira exaustiva. Não deixe de ler o material de referência da disciplina.

Para reduzir o número de linhas de código, os exemplos apresentados omitem propositalmente a importação das classes. Para esta aula, a maior parte das classes pertencem ao pacote **java.sql**. As principais IDEs Java possuem recursos para auxiliar a inclusão das importações das classes.

JDBC API

- Permite que programas Java se conectem a uma **base de dados** para efetuar consultas e alterações utilizando *Structured Query Language* (SQL).
- O programa pode se conectar a bases de dados de diversos fornecedores através de **drivers** específicos para cada base.

Usos típicos de JDBC

- Modelo cliente/servidor com um cliente GUI e a base de dados em um servidor.
- Modelo em três camadas, onde o cliente é responsável pela apresentação visual e há um servidor (*middle tier*) com os componentes responsáveis pela lógica de negócios, que fazem o acesso à base de dados localizado em outro servidor. Neste caso, a comunicação entre o cliente e o servidor *middle tier* utiliza algum protocolo como HTTP ou RMI.

Bases de dados relacionais

- Organizam os dados em relações (também chamadas de listas ou **tabelas**).
- As tabelas são composta por **linhas** (registros) e são definidas pelos nomes e tipos das **colunas** (campos).

Structured Query Language

- Linguagem para execução de comandos nas bases de dados relacionais.

Exemplo



tabela **contas**

nro_conta	saldo
123	500,00
124	1000,00
125	2500,00

tabela **titulares**

nro_titular	nome	rg	cpf
2	Marcos Antônio	22333444	09988877765
3	Rosimeire Aparecida	11222333	08866655543
4	Roberto Carlos	33444555	07755544432
5	André Barros	44555666	06655533321

tabela **contas_titulares**

nro_conta	nro_titular
123	2
124	3
124	5
125	4

Criação da tabela **contas** e inserção de dados iniciais

```
CREATE TABLE contas (  
    nro_conta BIGINT NOT NULL,  
    saldo DECIMAL(8,2),  
    PRIMARY KEY (nro_conta)  
)  
  
INSERT INTO contas VALUES (123, 500.00);  
INSERT INTO contas VALUES (124, 1000.00);  
INSERT INTO contas VALUES (125, 2500.00);
```

Criação da tabela **titulares** e inserção de dados iniciais

```
CREATE TABLE titulares (  
    nro_titular BIGINT NOT NULL,  
    nome VARCHAR(255) NOT NULL,  
    rg VARCHAR(32) NOT NULL,  
    cpf VARCHAR(32) NOT NULL,  
    PRIMARY KEY (nro_titular)  
)  
  
INSERT INTO titulares VALUES (2, 'Marcos Antônio', '22333444', '09988877765');  
INSERT INTO titulares VALUES (3, 'Rosimeire Aparecida', '11222333', '08866655543');  
INSERT INTO titulares VALUES (4, 'Roberto Carlos', '33444555', '07755544432');  
INSERT INTO titulares VALUES (5, 'André Barros', '44555666', '06655533321');
```

Criação da tabela **contas_titulares** e inserção de dados iniciais

```
CREATE TABLE contas_titulares (  
    nro_conta BIGINT NOT NULL,  
    nro_titular BIGINT NOT NULL,  
    PRIMARY KEY (nro_conta,nro_titular)  
);  
  
INSERT INTO contas_titulares VALUES (123,2);  
INSERT INTO contas_titulares VALUES (124,3);  
INSERT INTO contas_titulares VALUES (124,5);  
INSERT INTO contas_titulares VALUES (125,4);
```

Consulta de todos os dados de todos os titulares do banco

```
SELECT * FROM titulares;
```

2	Marcos Antônio	22333444	09988877765
3	Rosimeire Aparecida	11222333	08866655543
4	Roberto Carlos	33444555	07755544432
5	André Barros	44555666	06655533321

Consulta de NOME e RG de todos os titulares do banco

```
SELECT nome, rg FROM titulares;
```

Marcos Antônio	22333444
Rosimeire Aparecida	11222333
Roberto Carlos	33444555
André Barros	44555666

Consulta de todos os dados de todas as contas do banco

```
SELECT * FROM contas;
```

123	500.00
124	1000.00
125	2500.00

Consulta dos números das contas com saldo maior que R\$ 750,00

```
SELECT nro_conta FROM contas WHERE saldo > 750;
```

124
125

Consulta dos dados dos titulares com 'Ro' no seu nome

```
SELECT * FROM titulares WHERE nome LIKE '%Ro%';
```

3	Rosimeire Aparecida	11222333	08866655543
4	Roberto Carlos	33444555	07755544432

Consulta de dados das tabelas contas, titulares e contas_titulares

```
SELECT * FROM contas, titulares, contas_titulares;
```

123	500.00	2	Marcos Antônio	22333444	09988877765	123	2
123	500.00	2	Marcos Antônio	22333444	09988877765	124	3
123	500.00	2	Marcos Antônio	22333444	09988877765	124	5
123	500.00	2	Marcos Antônio	22333444	09988877765	125	4
123	500.00	3	Rosimeire Aparecida	11222333	08866655543	123	2
123	500.00	3	Rosimeire Aparecida	11222333	08866655543	124	3
123	500.00	3	Rosimeire Aparecida	11222333	08866655543	124	5
123	500.00	3	Rosimeire Aparecida	11222333	08866655543	125	4
123	500.00	4	Roberto Carlos	33444555	07755544432	123	2
123	500.00	4	Roberto Carlos	33444555	07755544432	124	3
123	500.00	4	Roberto Carlos	33444555	07755544432	124	5
123	500.00	4	Roberto Carlos	33444555	07755544432	125	4
123	500.00	5	André Barros	44555666	06655533321	123	2
123	500.00	5	André Barros	44555666	06655533321	124	3
123	500.00	5	André Barros	44555666	06655533321	124	5
123	500.00	5	André Barros	44555666	06655533321	125	4
124	1000.00	2	Marcos Antônio	22333444	09988877765	123	2
124	1000.00	2	Marcos Antônio	22333444	09988877765	124	3
124	1000.00	2	Marcos Antônio	22333444	09988877765	124	5
124	1000.00	2	Marcos Antônio	22333444	09988877765	125	4

Consulta de dados das tabelas contas, titulares e contas_titulares (cont.)

124	1000.00	3	Rosimeire Aparecida	11222333	08866655543	123	2
124	1000.00	3	Rosimeire Aparecida	11222333	08866655543	124	3
124	1000.00	3	Rosimeire Aparecida	11222333	08866655543	124	5
124	1000.00	3	Rosimeire Aparecida	11222333	08866655543	125	4
124	1000.00	4	Roberto Carlos	33444555	07755544432	123	2
124	1000.00	4	Roberto Carlos	33444555	07755544432	124	3
124	1000.00	4	Roberto Carlos	33444555	07755544432	124	5
124	1000.00	4	Roberto Carlos	33444555	07755544432	125	4
124	1000.00	5	André Barros	44555666	06655533321	123	2
124	1000.00	5	André Barros	44555666	06655533321	124	3
124	1000.00	5	André Barros	44555666	06655533321	124	5
124	1000.00	5	André Barros	44555666	06655533321	125	4
125	2500.00	2	Marcos Antônio	22333444	09988877765	123	2
125	2500.00	2	Marcos Antônio	22333444	09988877765	124	3
125	2500.00	2	Marcos Antônio	22333444	09988877765	124	5
125	2500.00	2	Marcos Antônio	22333444	09988877765	125	4
125	2500.00	3	Rosimeire Aparecida	11222333	08866655543	123	2
125	2500.00	3	Rosimeire Aparecida	11222333	08866655543	124	3
125	2500.00	3	Rosimeire Aparecida	11222333	08866655543	124	5
125	2500.00	3	Rosimeire Aparecida	11222333	08866655543	125	4

Consulta de dados das tabelas contas, titulares e contas_titulares (cont.)

125	2500.00	4	Roberto Carlos	33444555	07755544432	123	2
125	2500.00	4	Roberto Carlos	33444555	07755544432	124	3
125	2500.00	4	Roberto Carlos	33444555	07755544432	124	5
125	2500.00	4	Roberto Carlos	33444555	07755544432	125	4
125	2500.00	5	André Barros	44555666	06655533321	123	2
125	2500.00	5	André Barros	44555666	06655533321	124	3
125	2500.00	5	André Barros	44555666	06655533321	124	5
125	2500.00	5	André Barros	44555666	06655533321	125	4

Consulta dos nomes dos titulares e seus respectivos saldos

```
SELECT nome,saldo FROM contas, titulares, contas_titulares
WHERE contas.nro_conta = contas_titulares.nro_conta
AND titulares.nro_titular = contas_titulares.nro_titular;
```

Marcos Antônio	500.00
Rosimeire Aparecida	1000.00
Roberto Carlos	2500.00
André Barros	1000.00

Dedução de R\$ 5,00 de todas as contas

```
UPDATE contas SET saldo = saldo - 5.00;  
SELECT * FROM contas;
```

123	495.00
124	995.00
125	2495.00

Inserção de nova conta

```
INSERT INTO contas VALUES (126, 4000.00);  
SELECT * FROM contas;
```

123	495.00
124	995.00
125	2495.00
126	4000.00

Remoção de um registro de contas_titulares

```
DELETE FROM contas_titulares WHERE nro_conta = 124 AND nro_titular = 3;  
SELECT * FROM contas_titulares;
```

123	2
124	5
125	4

Utilização de JDBC

Carregar driver JDBC

O driver é disponibilizado pelo fornecedor da base de dados.

Exemplo de código para carregar o driver JDBC

```
try {  
    // Class.forName("oracle.jdbc.driver.OracleDriver");  
    Class.forName("org.apache.derby.jdbc.ClientDriver");  
} catch(ClassNotFoundException e) {  
    System.err.println("erro carregando o driver " + e);  
}
```

Definir a URL da conexão

É necessário especificar o host, a porta e o nome da base de dados.

O formato é definido na documentação de cada driver.

Exemplos:

```
String host = "localhost";  
int port = 1527;  
String dbName = "sample";  
String derbyURL;  
derbyURL = "jdbc:derby://" + host + ":" + port + "/" + dbName;  
String oracleURL;  
oracleURL = "jdbc:oracle:thin:@" + host + ":" + port + ":" + dbName;
```

Estabelecer a conexão

Exemplo:

```
String username = "app";  
String password = "app";  
try {  
    Connection connection;  
    connection = DriverManager.getConnection(derbyURL, username, password);  
    System.out.println("Conexão estabelecida!");  
} catch (SQLException e) {  
    System.err.println("erro estabelecendo a conexão: " + e);  
}
```

Métodos da classe Connection bastante utilizados:

- createStatement
- prepareStatement
- prepareCall
- rollback/commit
- close
- isClosed

Criar uma instância da classe Statement

A criação é feita a partir da instância de Connection:

```
Statement statement;  
statement = connection.createStatement();
```

Executar uma consulta ou uma alteração nos dados utilizando um dos métodos de Statement

Exemplo:

```
String query;  
query = "SELECT customer_id, name, city FROM customer";  
ResultSet result = statement.executeQuery(query);
```

Outros métodos da classes Statement:

- executeQuery
- executeUpdate
- executeBatch
- setQueryTimeout
- getMaxRows/setMaxRows

Processar os resultados, que são devolvidos em uma instância de ResultSet (no caso de uma consulta)

A forma mais simples de acessar o resultado é utilizar os métodos da instância de ResultSet, tais como:

- next
- getInt
- getString

Os métodos que seguem o padrão getXxx permitem referenciar a coluna por um índice (com contagem iniciando em 1) ou pelo nome do campo.

Exemplo:

```
ResultSet result = statement.executeQuery(query);
while (result.next()) {
    System.out.println(result.getInt(1) + " : "
        + result.getString(2) + " : " + result.getString("city") );
}
```

Fechar a conexão.

O fechamento da conexão é feito usando o método `close` de `Connection`:

```
connection.close();
```

Esta operação deve ser adiada caso haja a expectativa de executar outras operações na base de dados.

Com a finalidade de reutilizar conexões existentes, a API JDBC 2.0 define uma interface *ConnectionPoolDataSource*.

Exemplo de aplicação que faz a consulta de todas as contas

```
public class AppSelectContas {  
    public static void main(String[] args)  
        throws ClassNotFoundException, SQLException {  
  
        Class.forName("org.apache.derby.jdbc.ClientDriver");  
        Connection conexao;  
        String url = "jdbc:derby://127.0.0.1:1527/banco";  
        String usuario = "app";  
        String senha = "app";  
        conexao = DriverManager.getConnection(url, usuario, senha);  
  
        Statement st;  
        st = conexao.createStatement();  
  
        String sql = "SELECT nro_conta, saldo FROM contas";  
        ResultSet resultados = st.executeQuery(sql);  
  
        System.out.println("Dados das contas:");  
        while (resultados.next()) {  
            System.out.print("Número: " + resultados.getLong(1) + " - ");  
            System.out.println("Saldo: R$ " + resultados.getBigDecimal("saldo"));  
        }  
  
        conexao.close();  
    }  
}
```

Obrigado!

