Servlet & JDBC

ALBERTO BELUSSI

ANNO ACCADEMICO 2010/2011

Servlet: interazione con un DBMS

In Java è possibile interagire con un DBMS attraverso l'uso della libreria JDBC (Java Database Connectivity).

http://java.sun.com/javase/technologies/database

http://www.oracle.com/technetwork/java/javase/tech/index-jsp-136101.html

JDBC API: fornisce un insieme di classi JAVA che consentono un accesso standardizzato a qualsiasi DBMS purché questo fornisca un driver JDBC.

Driver JDBC

E’ un modulo software in grado di interagire con un DBMS. Traduce ogni invocazione dei metodi delle classi JDBC in comandi SQL accettati dal DBMS a cui è dedicato.

7 passi per interagire con un DBMS

1° PASSO

Caricare il driver JDBC per il DBMS che si utilizza.

- Per caricare il driver, basta caricare la classe corrispondente:

```
import java.sql.*
…
Class.forName("NomeDriver");
```

Per postgresql il nome del driver è:

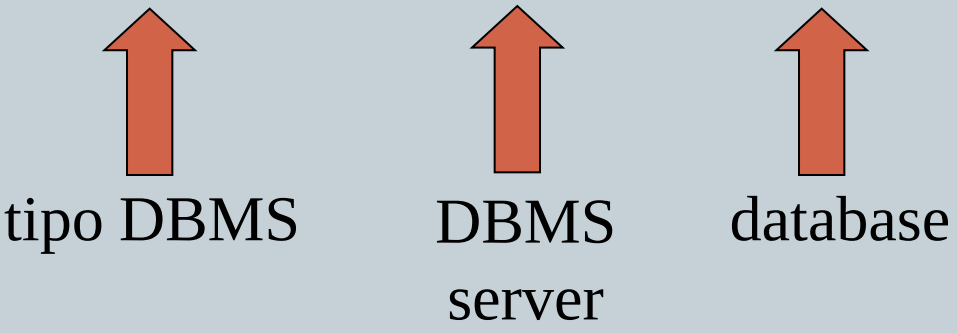
```
org.postgresql.Driver
```

7 passi per interagire con un DBMS

2° PASSO

Definire una connessione identificando l'URL della base di dati a cui ci si vuole connettere.

```
String URL =
  "jdbc:postgresql://dbserver/did2011"
```



7 passi per interagire con un DBMS

3° PASSO

Stabilire una connessione istanziando un oggetto della classe `Connection`:

```
String user = "Utente-postgres";
String passwd = "Password-utente";
Connection con = DriverManager.getConnection(
                    URL, user, passwd);
```

7 passi per interagire con un DBMS

4° PASSO

Utilizzando l'oggetto della classe `Connection` creato al passo precedente, creare un oggetto della classe `Statement` per poter sottomettere comandi al DBMS:

```
Statement stat = con.createStatement();
```

7 passi per interagire con un DBMS

5° PASSO

Eseguire un'interrogazione SQL o un comando SQL di aggiornamento di una tabella:

- Interrogazione

```
String query = "SELECT * FROM PERSONA";
ResultSet res = stat.executeQuery(query);
```

- Aggiornamento

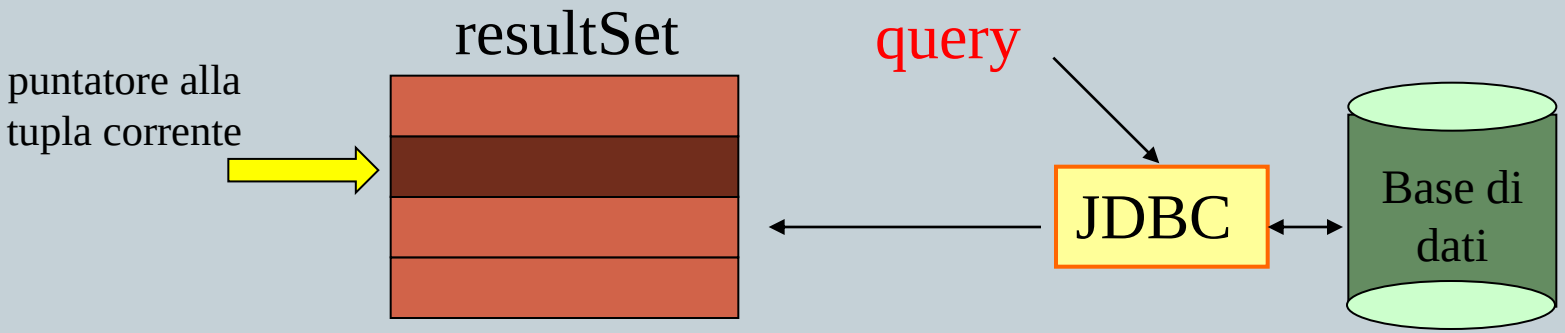
```
String update = "UPDATE PERSONA"+
                "SET NOME = 'Rosa' WHERE id = 1";
stat.executeUpdate(update);
```

7 passi per interagire con un DBMS

6° PASSO

Processare il risultato dell'interrogazione attraverso l'applicazione dei metodi della classe `resultSet`.

Il resultSet è un cursore che consente di accedere alle tuple risultato dell'interrogazione:



7 passi per interagire con un DBMS

6° PASSO

E' possibile scandire in modo sequenziale il contenuto di un resultSet:

```
String query = "SELECT * FROM PERSONA";
resultSet res = stat.executeQuery(query);
while(res.next()) {
        ……
}
```

Il metodo `next()` sposta il puntatore sulla tupla successiva.

7 passi per interagire con un DBMS

6° PASSO

E' possibile accedere alle proprietà della **tupla corrente** di un resultSet con i seguenti metodi:

`getXxxx(par)`: dove `Xxxx` è un tipo base di Java e `par` può essere un indice di posizione o il nome di un attributo della relazione risultato dell'interrogazione; questo metodo restituisce il valore in posizione `par` oppure il valore dell'attributo di nome `par` della tupla corrente.

`wasNull`: si riferisce all'ultima invocazione di `getXxxx` e restituisce true se il valore letto era uguale al valore nullo.

... (molti altri metodi sono disponibili)

7 passi per interagire con un DBMS

7° PASSO

Chiudere la connessione usando l'oggetto della classe `Connection`:

```
con.close();
```

7 passi per interagire con un DBMS

Variante del 5° PASSO

E' possibile ottimizzare l'esecuzione di una interrogazione che deve essere rifatta più volte usando la classe `PreparedStatement`.

Tale classe consente di inserire parametri nell'interrogazione e di valorizzarli, attraverso specifici metodi, prima dell'effettiva esecuzione dell'interrogazione stessa.

7 passi per interagire con un DBMS

Variante del 5° PASSO

Esempio:

```
Connection con = DriverManager.getConnection(URL,
    user, passwd);
String q = "SELECT Nome, Cognome"+
          " FROM Persona"+
          " WHERE id = ?";
PreparedStatement pstat = con.prepareStatement(q);
pstat.setInt(1, 15);
ResultSet res = pstat.executeQuery();
```

Transazioni in JDBC

E' possibile eseguire transazioni in JDBC come segue:

```
…
con.setAutoCommit(false);
preparedStatement ps1 = con.prepareStatement(
    "UPDATE CONTO SET SALDO=SALDO+?"+
    " WHERE NUMERO=? AND FILIALE = 'X'");
ps1.setInt(1, 1000); ps1.setInt(2, 358);
ps1.execute();
preparedStatement ps2 = con.prepareStatement(
    "UPDATE CONTO SET SALDO=SALDO-?"+
    " WHERE NUMERO=? AND FILIALE = 'X'");
ps2.setInt(1, 1000); ps2.setInt(2, 876);
ps2.execute();
con.commit();
con.setAutoCommit(true);
```