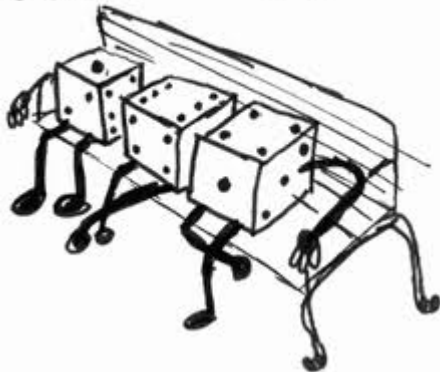




## Unidade 7

# Conectividade com Banco de Dados

O BANCO DE DADOS



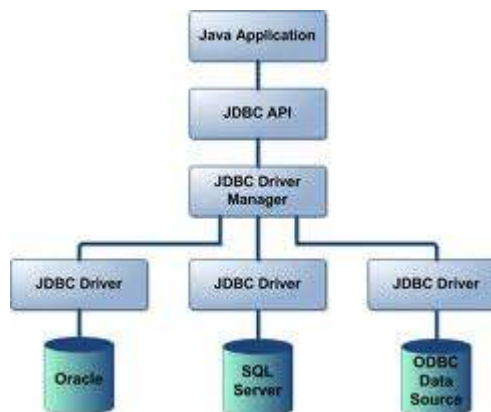
Prof. Aparecido V. de Freitas  
Doutor em Engenharia  
da Computação pela EPUSP





# Introdução

- Em 1996, a Sun liberou a primeira versão do JDBC – Java Database Connectivity.
- Esta liberação permitiu que programadores Java pudessem fazer conexão a um banco de dados, atualização e consultas através da linguagem SQL.





# Conectividade JDBC

- Programas desenvolvidos com Java e JDBC são independentes de plataforma e de fornecedores de SGBD.
- O mesmo programa Java pode rodar em um PC, uma workstation, etc.
- Você pode mover dados de um SGBD para outro (por exemplo, SQL Server para DB/2).





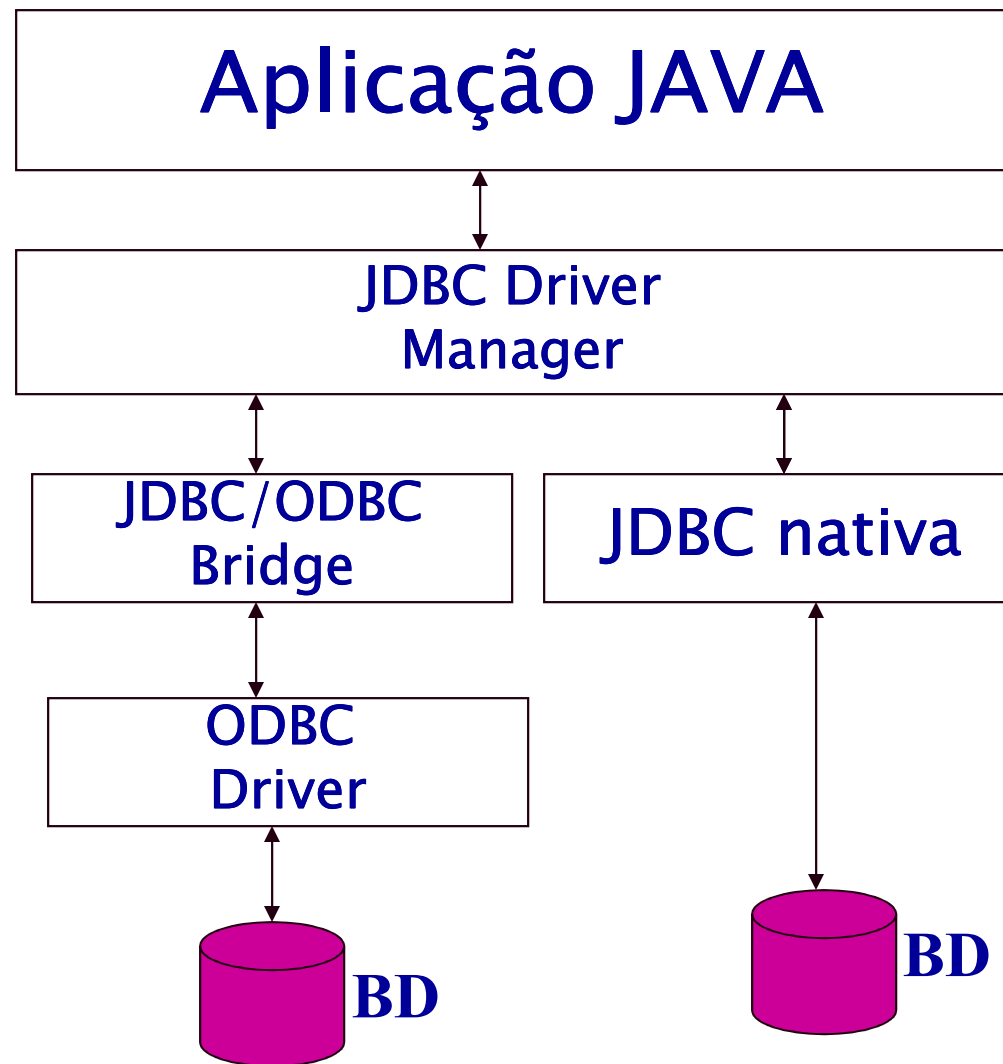
# Arquitetura JDBC

- Consiste de 2 camadas.
- A camada superior corresponde às API's JDBC.
- Esta camada se comunica com o JDBC driver manager por meio de comandos SQL.





# Arquitetura JDBC





# Objetivos JDBC

- Permitir que programadores Java possam escrever aplicações para acessar qualquer banco de dados.
- Permitir que fornecedores de SGBD possam fornecer drivers e otimizar estas liberações.





# Bridge ODBC-JDBC

- Alguns fornecedores de Bancos de dados têm drivers JDBC nativos e assim você poderá instalá-los, seguindo as orientações do fabricante.
- Uma vez que ODBC existe para a maioria dos fabricantes de Bancos de Dados, JavaSoft decidiu escrever uma bridge JDBC-ODBC.





# Bridge ODBC-JDBC

- Tem a vantagem de permitir que as pessoas usem JDBC imediatamente.
- Tem a desvantagem de requerer uma outra camada entre o banco de dados e JDBC.







# Conceitos JDBC

- A programação com classes JDBC não é diferente da programação Java usual.
- Você basicamente constrói objetos a partir de classes, extendendo-os por **herança** se necessário.





# Conexão ao Banco de Dados

- Para fazer a conexão ao banco de dados você deve especificar o “data-source” e pode também especificar parâmetros adicionais.
- Por exemplo, drivers de protocolos de rede podem necessitar de uma porta e drivers ODBC podem necessitar de atributos extras.





# Conexão ao banco de dados

- JDBC usa a seguinte sintaxe para descrever data-sources:

jdbc:odbc:datasource

- Esta especificação possibilita o acesso a um datasource ODBC, usando a bridge ODBC-JDBC.





# Conexão ao banco de dados

- A classe **DriverManager** é responsável pela carga dos drivers JDBC e cria uma nova conexão ao banco de dados.
- Você necessita carregar o driver do banco de dados para que sua aplicação possa fazer a comunicação com o banco de dados.





# Conexão ao banco de dados

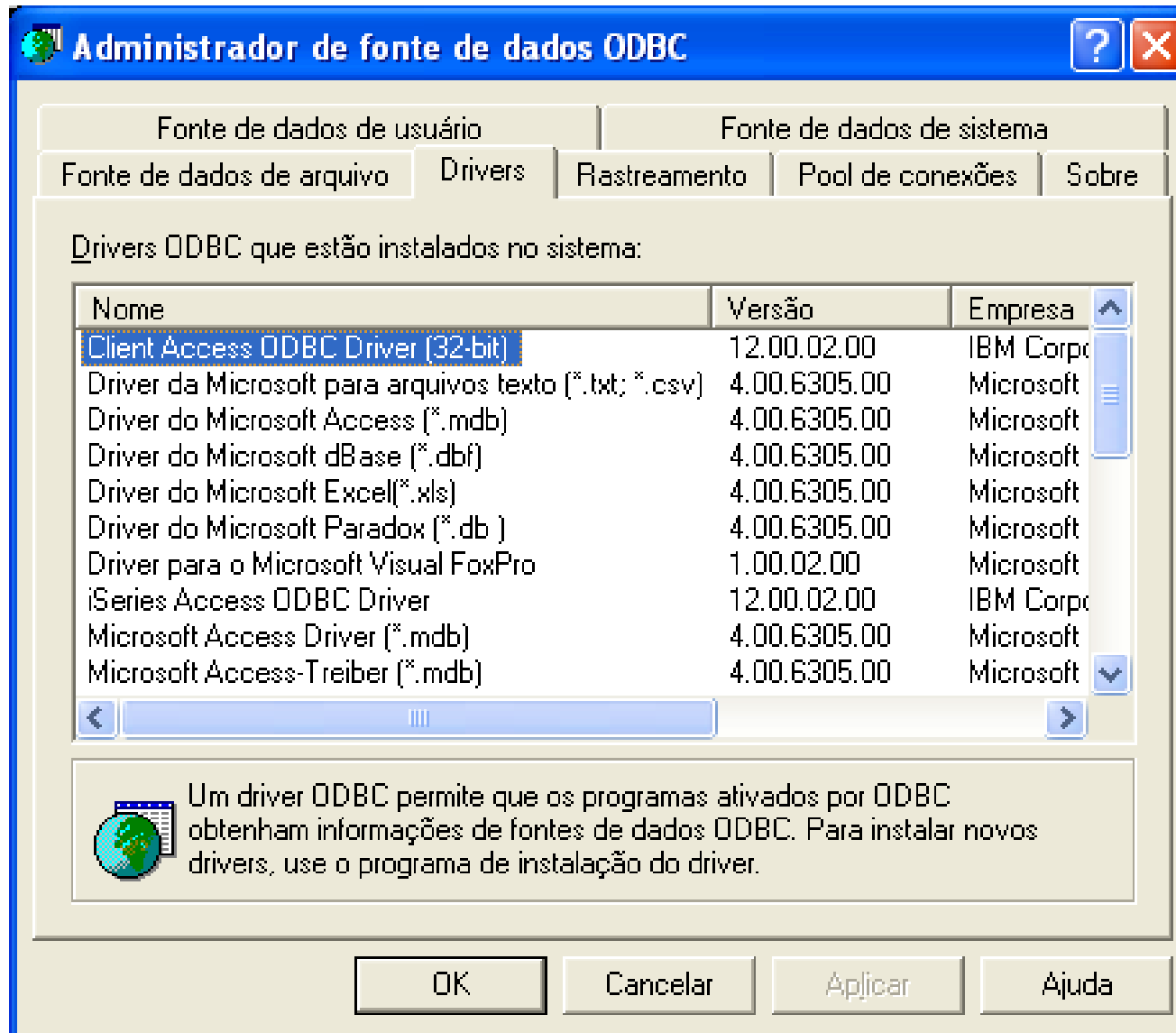
```
Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
```

Obs.: Se você tem o driver de algum fornecedor, então verifique na documentação qual é o nome do driver e carregue-o.





# Fonte de Dados





# Abrindo conexão

- Após a carga do driver, você deverá abrir uma conexão com o banco de dados, por meio do código:

```
String url = "jdbc:odbc:datasource";  
String user = "roberto";  
String password = "xyz98";  
Connection con =  
    DriverManager.getConnection(url,user,pwd) ;
```





# Abrindo conexão

- O gerenciador JDBC irá tentar encontrar um driver que pode usar o protocolo especificado na variável que define a fonte de dados.
- O gerenciador JDBC irá interagir com os drivers disponíveis correntemente registrados com o device manager.







# Executando queries

- O objeto conexão que você obtém através de uma chamada à função **getConnection** permite a você usar os drivers JDBC para executar queries SQL.
- Você pode executar **queries** e comandos de atualização, commit ou transações roolback.





# Executando queries

- ⊕ Para fazer uma query, você primeiro deverá criar um objeto do tipo Statement.
- ⊕ O objeto Conexão que você obtém da chamada `DriverManager.getConnection` pode criar objetos statements.

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





# Executando queries

- Comandos SQL podem ser executados pela chamada do método **stmt.executeUpdate(cmd)**.
- Uma query pode também ser executada pela chamada do método **executeQuery()** da classe **Statement**.
- O método **executeQuery()** retorna um objeto do tipo **ResultSet**.





# ResultSet

```
ResultSet rs =  
    stmt.executeQuery("SELECT *  
    FROM BOOKS");  
  
while (rs.next() )  
    {  
        acesse a linha do result set  
    }
```





## Acessando o conteúdo do ResultSet

- Durante o acesso de uma linha individual, certamente você irá querer acessar o conteúdo da linha.
- Existem métodos para isso.

```
String isbn = rs.getString(1);  
float price = rs.getDouble("Price");
```





# Acessores para tipos Java

- ◆ Correspondem às funções para acesso ao ResultSet.
- ◆ Cada acessor tem duas formas. Uma forma tem um argumento numérico e outra com argumento String.
- ◆ Quando você fornece um argumento numérico, você está se referindo à coluna que corresponde àquele número.
- ◆ Quando você fornece um argumento String você se refere à coluna cujo nome corresponde ao String fornecido.





# Acessores para tipos Java

- `rs.getString(1)` retorna o valor da primeira coluna na linha corrente.
- `rs.getDouble("Price")` retorna o valor da coluna com nome "Price".





# Atividade

Escrever um programa **desktop** que faça uma conexão a um banco de dados e insira um registro. Utilizar a bridge para conexão JDBC/ODBC.

**Acessar o Servidor de Banco de Dados MySQL.**

- Obs.    a) Nome do database: CURSO  
         b) Nome da tabela: TABCURSO

Código do Curso	Nome do Curso
CODCURSO int(2)	NOMECURSO char(50)







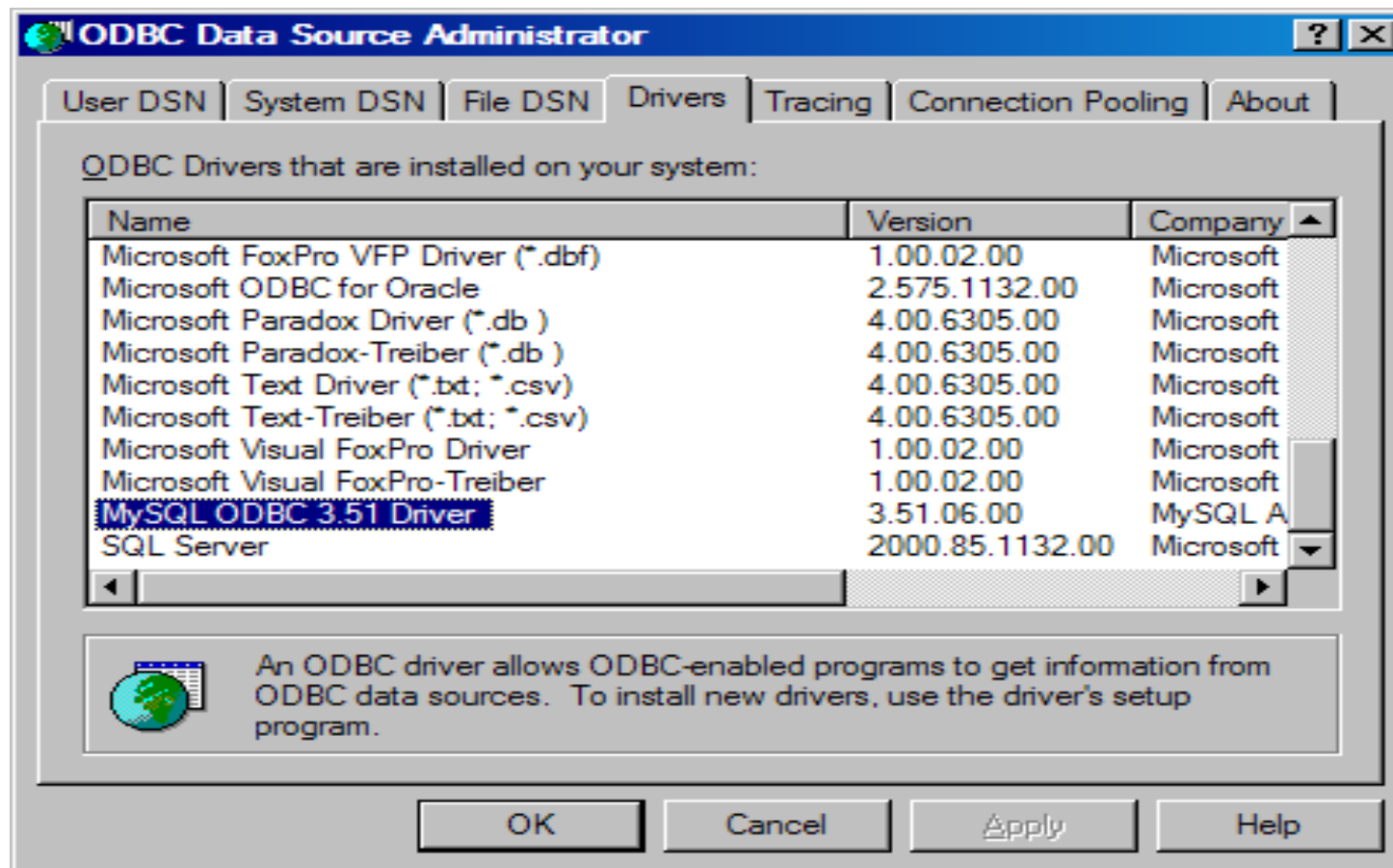
## Criação do Banco de Dados – MySQL

```
mysql> create database curso;
mysql> use curso;
mysql> show tables;
mysql> create table tabcurso (
                codcurso      int(2) unsigned    zerofill not null,
                nomecurso      char(50),
                primary key (codcurso) ) ;
mysql> describe tabcurso;
mysql> show tables;
mysql> insert into tabcurso values(1,'Matematica');
mysql> select * from tabcurso;
```

Configuração ODBC => Comando

**ODBCAD32 (Configurar Fonte de  
Dados de Usuário**







**MySQL ODBC 3.51 Driver - DSN Configuration, Version 3.51.06**

This dialog helps you in configuring the ODBC Data Source Name, that you can use to connect to MySQL server

**DSN Information**

Data Source Name:

Description:

**MySQL Connection Parameters**

Host/Server Name(or IP):


Database Name:

User:

Password:

Port (if not 3306):

SQL command on connect:





```
import java.sql.Connection;  
import java.sql.DriverManager;  
import java.sql.SQLException;  
import java.sql.Statement;
```

```
public class Atividade07 { // Conexão com MySQL
```

```
public static void main(String[] args) {
```

```
    Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
```

```
        String url = "jdbc:odbc:curso";
```

```
        Connection con = DriverManager.getConnection(url, "root", null);
```

```
        System.out.println("\nConexao no Servidor MySQL feita com sucesso...");
```

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

```
        String command = "INSERT INTO tabcurso VALUES(1,'Psicologia')";
```

```
        stmt.executeUpdate(command);
```

```
        System.out.println("\nGravacao no Banco de Dados feita com sucesso...");
```

```
}
```





```
catch (SQLException ex) {  
  
    System.out.println ("**** ERRO DE ACESSO AO BANCO DE DADOS...|n");  
    System.out.println ("****SQLException: " + ex);  
  
}  
  
catch (Exception ex) {  
  
    System.out.println("*****Exception: " + ex);  
  
}  
  
}  
  
}
```





# Atividade

Escrever um programa **desktop** que faça uma conexão a um banco de dados e insira um registro. Utilizar a bridge para conexão JDBC/ODBC.

**Acessar o Servidor de Banco de Dados DB2 da plataforma IBM i.**

- Obs.    a) Nome do database: CURSO  
          b) Nome da tabela: TABCURSO

Código do Curso	Nome do Curso
CODCURSO char(2)	NOMECURSO char(50)





package `uscs`;

```
import java.sql.Connection;  
import java.sql.DriverManager;  
import java.sql.SQLException;  
import java.sql.Statement;
```

```
public class Atividade08 { // Conexão DB2 – IBM i
```

```
public static void main(String[] args) {
```

```
try {
```

```
    Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
```

```
String url = "jdbc:odbc:tabcurso";
```

```
Connection con = DriverManager.getConnection(url, "db2uscs", "uscs");
```

```
System.out.println("\nConexao no IBM i feita com sucesso...");
```

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

```
String command = "INSERT INTO tabcurso VALUES(1,'Psicologia')";
```

```
stmt.executeUpdate(command);
```

```
System.out.println("\nGravacao no Banco de Dados feita com sucesso...");
```

```
}
```





```
catch (SQLException ex) {  
  
    System.out.println ("**** ERRO DE ACESSO AO BANCO DE DADOS...|n");  
    System.out.println ("****SQLException: " + ex);  
  
}  
  
catch (Exception ex) {  
  
    System.out.println("*****Exception: " + ex);  
  
}  
  
}  
  
}
```







# Atividade

Escrever um programa **desktop** que faça uma conexão a um banco de dados e insira um registro. Utilizar a bridge para conexão JDBC/ODBC.

**Acessar o Servidor de Banco de Dados MySQL. Utilizar o Driver JDBC nativo.**

- Obs.    a) Nome do database: CURSO  
         b) Nome da tabela: TABCURSO

Código do Curso	Nome do Curso
CODCURSO int(2)	NOMECURSO char(50)





## Driver JDBC MySQL nativo

- ✓ Baixar o driver a partir do endereço:

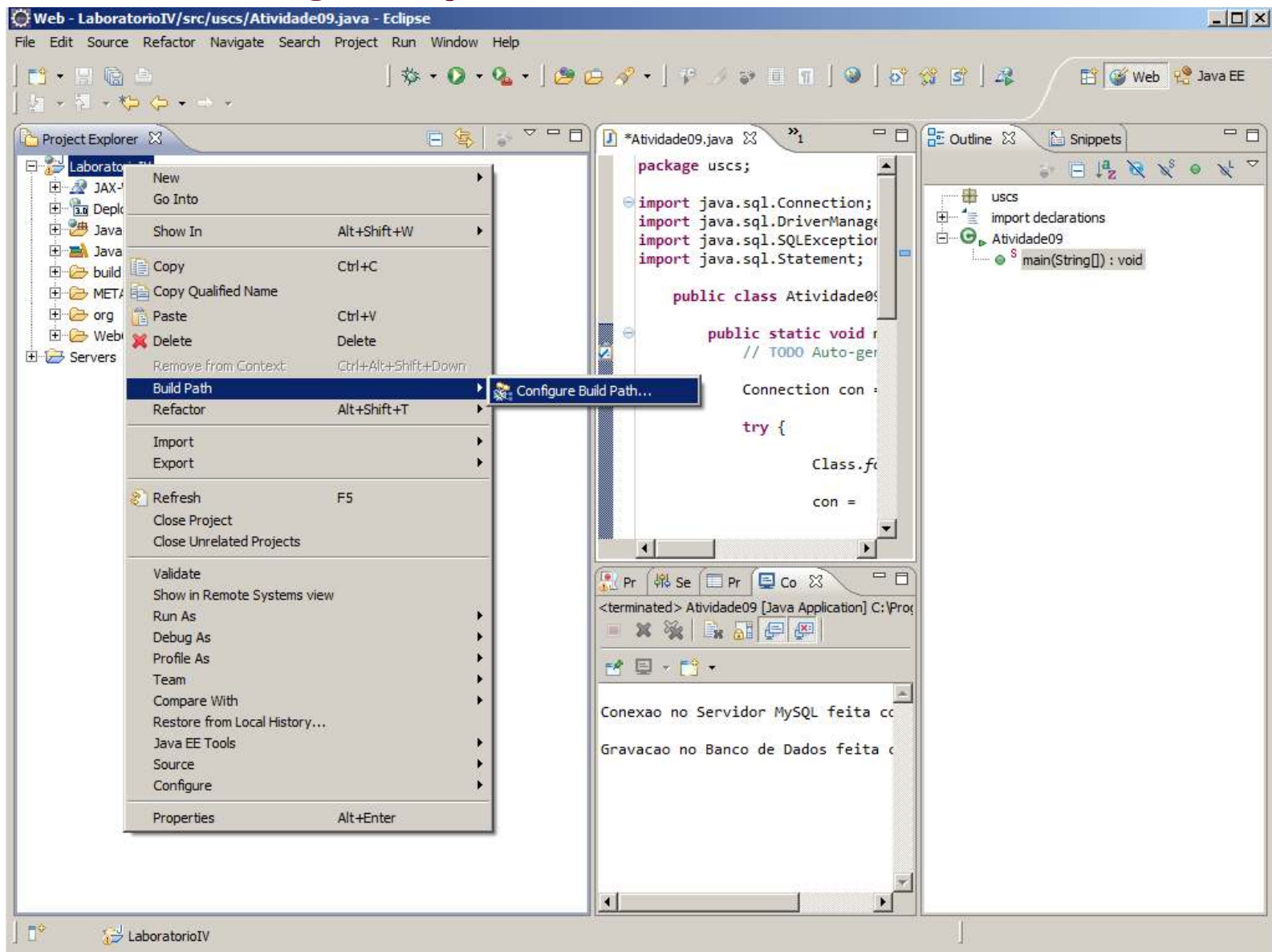
**<http://dev.mysql.com/downloads/connector/j/>**

- ✓ Salvar em algum diretório do Servidor
- ✓ Configurar o Path do Eclipse para que o projeto visualize o driver



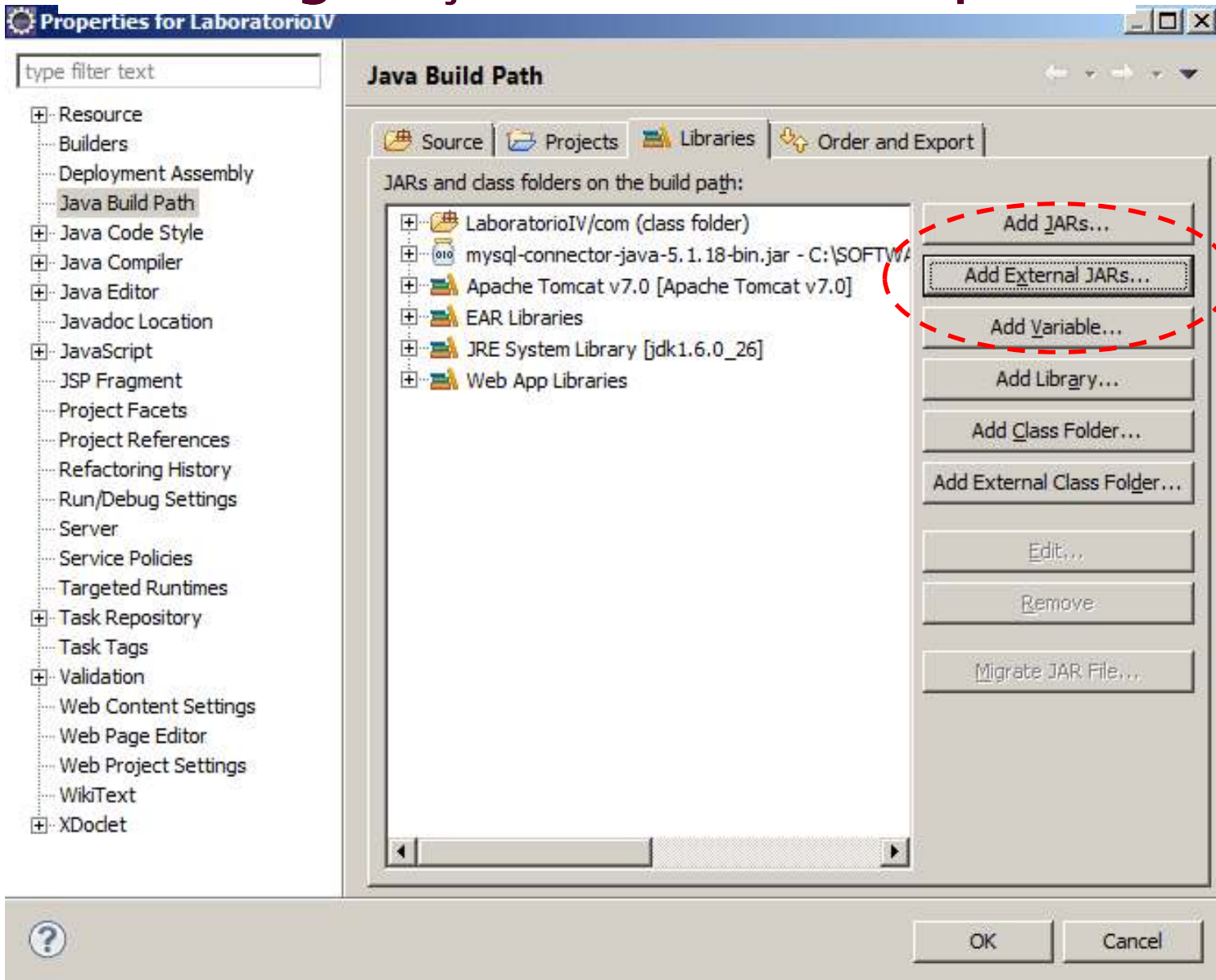


# Configuração do Path – Eclipse





# Configuração do Path – Eclipse





## Parâmetros de Conexão

```
Connection con = null;
```

```
Class.forName("com.mysql.jdbc.Driver").newInstance();
```

```
con = DriverManager.getConnection("jdbc:mysql://localhost/curso?" +  
    " user=root&password="+ "" );
```





package uscs;

```
import java.sql.Connection;  
import java.sql.DriverManager;  
import java.sql.SQLException;  
import java.sql.Statement;
```

```
public class Atividade09 {
```

```
    public static void main(String[] args) {  
        // TODO Auto-generated method stub
```

```
        Connection con = null;
```

```
        try {
```

```
            Class.forName(" com.mysql.jdbc.Driver ").newInstance();
```

```
            con =
```

```
                DriverManager.getConnection( "jdbc:mysql://localhost/curso?" +  
                    "user=root&password="+ "" );
```

```
            System.out.println("\nConexao no Servidor MySQL feita com sucesso...");
```





```
Statement stmt = con.createStatement();
String command = "INSERT INTO tabcurso VALUES(2,'Matematica')" ;

stmt.executeUpdate(command);
System.out.println("\nGravacao no Banco de Dados feita com sucesso...");

}
catch (SQLException ex) {

    System.out.println ("**** ERRO DE ACESSO AO BANCO DE DADOS...\n");
    System.out.println ("****SQLException: " + ex);

}

catch (Exception ex) {

    System.out.println("*****Exception: " + ex);

}

}

}
```

