UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

NGÔN NGỮ LẬP TRÌNH JAVA

CHƯƠNG 7: LẬP TRÌNH CSDL VỚI JDBC (P1)

GVGD: ThS. Lê Thanh Trọng

Nội dung

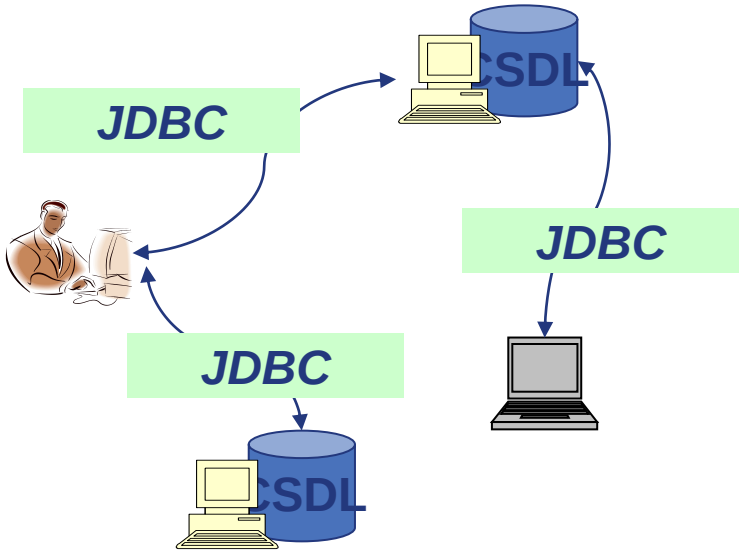
- Mục tiêu bài học
- Giới thiệu JDBC
- JDBC Drivers
- Lớp DriverMananger
- Các bước kết nối CSDL
- Đăng kí driver và kết nối CSDL
- Tóm tắt bài học

Mục tiêu bài học

- Giới thiệu về lập trình CSDL trong Java với JDBC API.
- Nhằm trang bị cho sinh viên các kiến thức về JDBC, các drivers, lớp DriverManagement.
- Sinh viên nắm được các bước chính trong việc kết nối CSDL và có thể tiến hành đăng kí và kết nối CSDL từ ứng dụng java cụ thể.

Giới thiệu về JDBC

- JDBC (Java DataBase Connectivity) là một thư viện chuẩn dùng để truy xuất các cơ sở dữ liệu dạng bảng như MS Acess, SQL Server, Oracle,… trong các ứng dụng Java bằng ngôn ngữ truy vấn SQL.
- Các hàm truy xuất cơ sở dữ liệu với JDBC nằm trong gói java.sql.*



Tại sao cần JDBC?

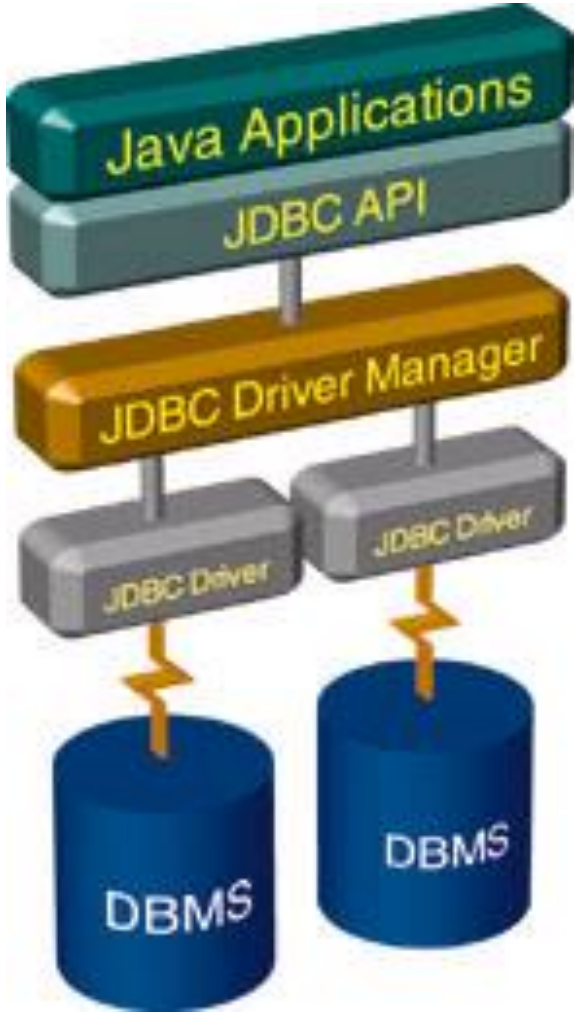
UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

- JDBC giúp các Java Developers tạo nên các ứng dụng truy xuất cơ sở dữ liệu mà không cần phải học và sử dụng các APIs do các công ty sản xuất phần mềm khác nhau bên thứ ba cung cấp.
- JDBC đảm bảo rằng bạn sẽ có thể phát triển nên các ứng dụng truy cập cơ sở dữ liệu có khả năng truy cập đến các RDBMS khác nhau bằng cách sử dụng các JDBC driver khác nhau.

Có thể làm gì với JDBC?

- Kết nối với nguồn dữ liệu (database)
- Gởi các các câu lệnh truy vấn, cập nhật dữ liệu đến csdl
- Truy vấn và xử lý kết quả

Kiến trúc JDBC

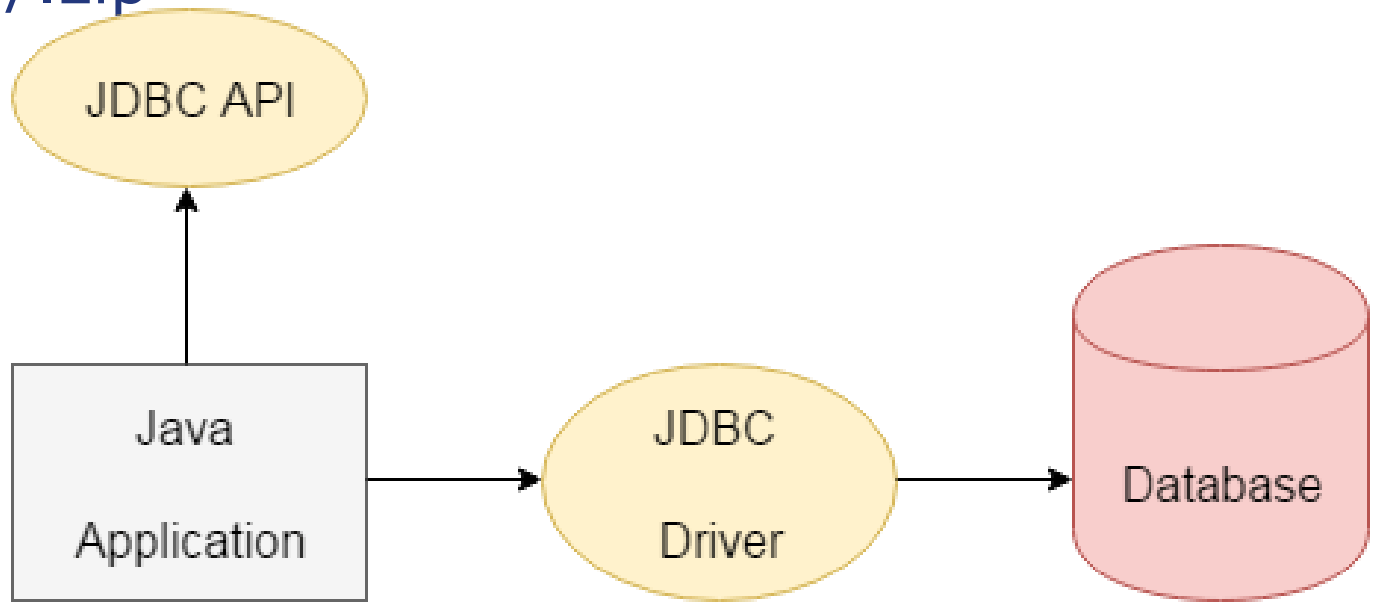


Các khái niệm cơ bản

- **JDBC API:** là một API hoàn toàn dựa trên Java.
- **JDBC DriverManager:** là trình quản lý JDBC giao tiếp trực tiếp với các trình điều khiển cơ sở dữ liệu cụ thể - giao tiếp thực sự với cơ sở dữ liệu.
- Các RDBMS hay các nhà sản xuất phần mềm thứ 3 phát triển các drivers cho java đều phải tuân thủ đặc tả JDBC.
- Các java developers dùng các JDBC drivers để phát triển các ứng dụng có truy cập, thao tác CSDL.

JDBC Drivers

- Tập các lớp giúp truy cập đến các hệ DBMS khác nhau dùng kỹ thuật JDBC
- Do các hãng xây dựng DBMS hoặc một đơn vị thứ 3 khác cung cấp.
- Gói trong .jar/.zip



Các loại JDBC Drivers

- Có 4 loại JDBC drivers:
  - JDBC-ODBC bridge driver
  - Native-API driver (partially java driver)
  - Network Protocol driver (fully java driver)
  - Thin driver (fully java driver)

JDBC-ODBC bridge driver

- The JDBC-ODBC bridge driver sử dụng các ODBC driver để kết nối đến CSDL. Nó chuyển các phương thức JDBC sang các lời gọi hàm ODBC.

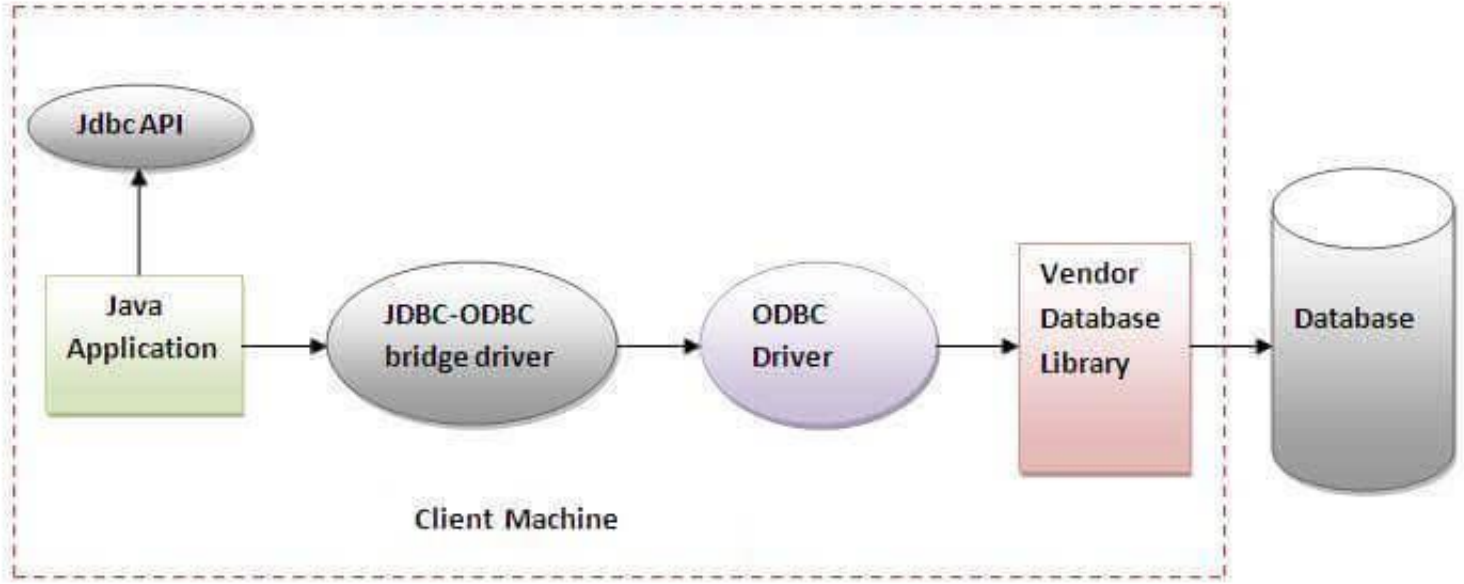


Figure- JDBC-ODBC Bridge Driver

JDBC-ODBC bridge driver

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

- Ưu điểm:
  - Dễ sử dụng.
  - Kết nối dễ dàng đến bất kì CSDL nào.
- Nhược điểm:
  - Khá chậm vì tốn tài nguyên chuyển đổi các phương thức JDBC sang ODBC
  - Các ODBC driver cần được cài đặt ở máy client.

Native-API driver

- The Native API driver sử dụng các thư viện client-side của database.
- Driver này chuyển đổi các JDBC method sang dạng native calls của database API.
- Không được viết hoàn toàn bằng java

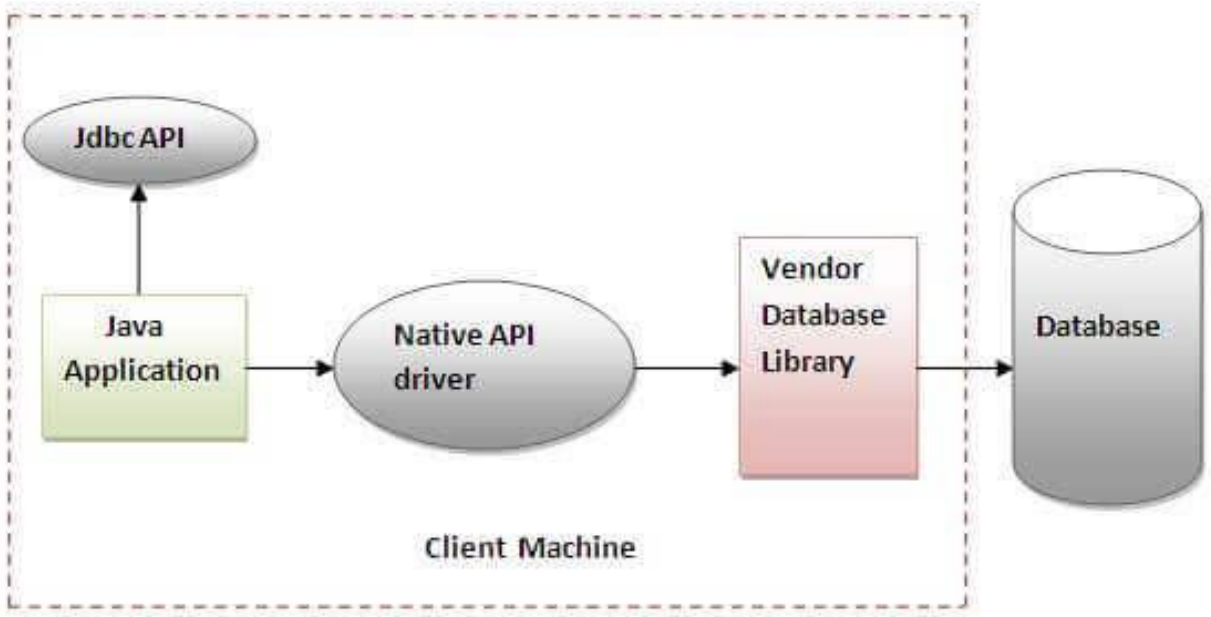


Figure- Native API Driver

Native-API driver

- Ưu:
  - Nhanh hơn so với JDBC-ODBC bridge driver.
- Nhược:
  - Native driver cần được cài trên từng máy client
  - Các native library hỗ trợ kết nối với DBMS cụ thể phải được cài đặt trên client

Network Protocol driver

- The Network Protocol driver sử dụng middleware (application server) để chuyển các JDBC calls trực tiếp hoặc gián tiếp thành các nghi thức DBMS đặc thù.
- Viết hoàn toàn bằng Java

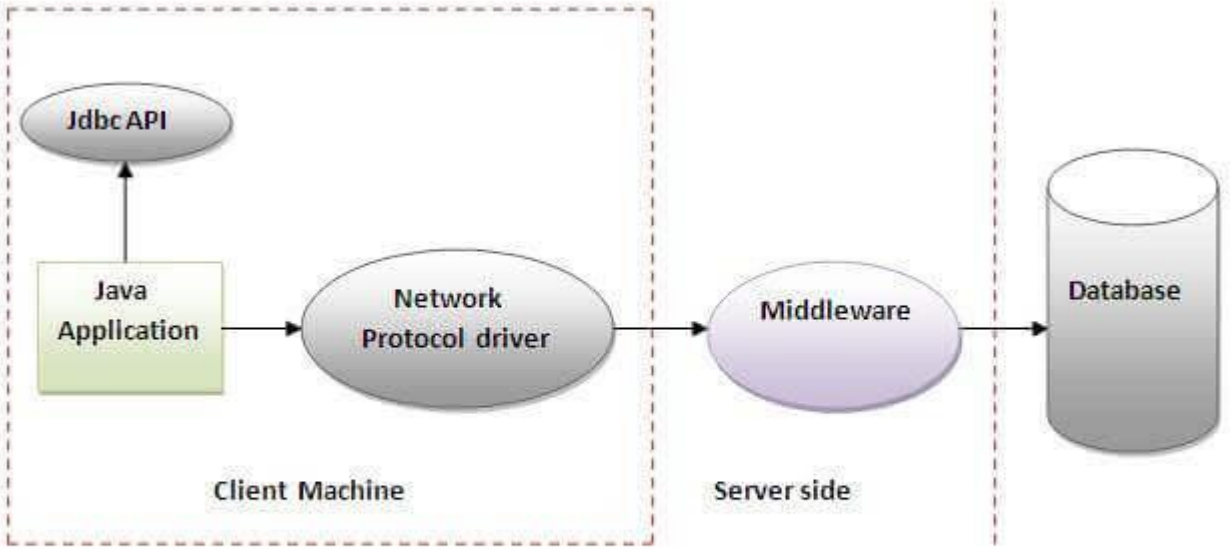


Figure- Network Protocol Driver

Network Protocol driver

- Ưu:
  - Không cần cài các thư viện driver trên máy client. Sự chuyển này đặt ở phía server mà không đòi hỏi cài đặt trên máy tính client
- Nhược:
  - Máy tính phải kết nối mạng.
  - Việc bảo trì các Network Protocol driver có thể khá lớn.

Thin driver

- Các thin driver chuyển JDBC calls trực tiếp sang giao thức DBMS cụ thể.
- Được viết hoàn toàn bằng Java.

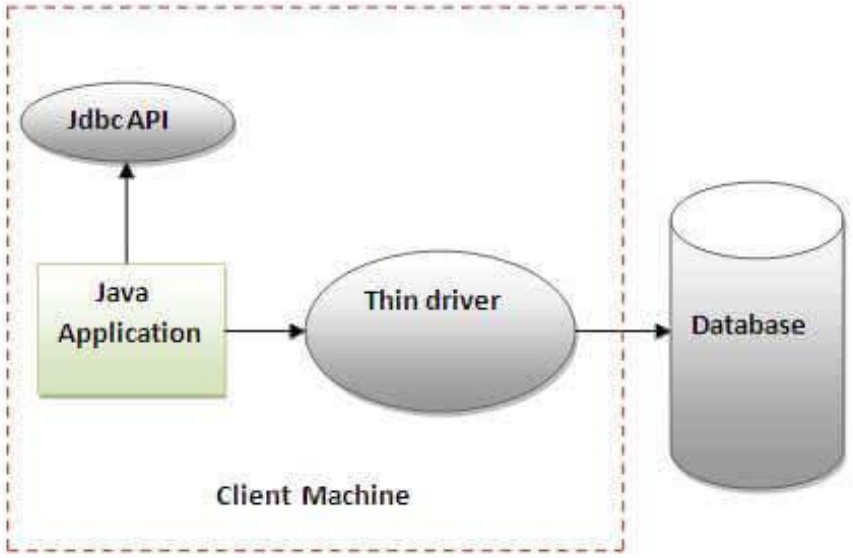


Figure- Thin Driver

Thin driver

- Ưu:
  - Nhanh hơn các drivers khác.
  - Khắc phục nhược điểm các loại drivers khác.
- Nhược:
  - Drivers phụ thuộc vào Database.

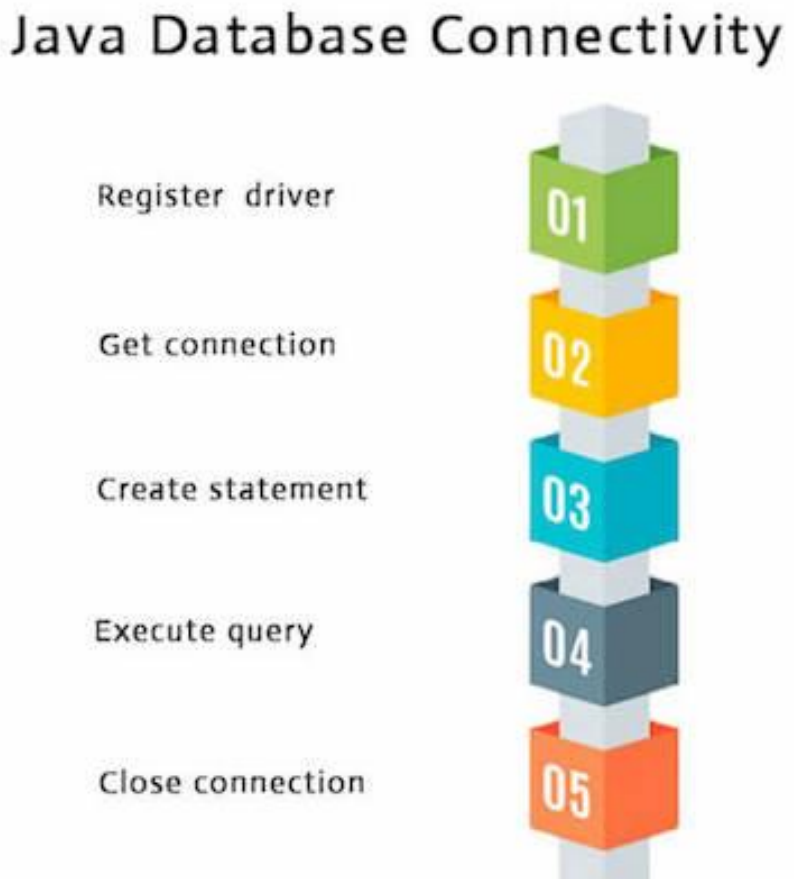
DriverManager

- DriverManager class đóng vai trò trung gian giao tiếp giữa người sử dụng và drivers.
- Nó lưu dấu các drivers sẵn có và thao tác thành lập các kết nói giữa database với driver phù hợp.

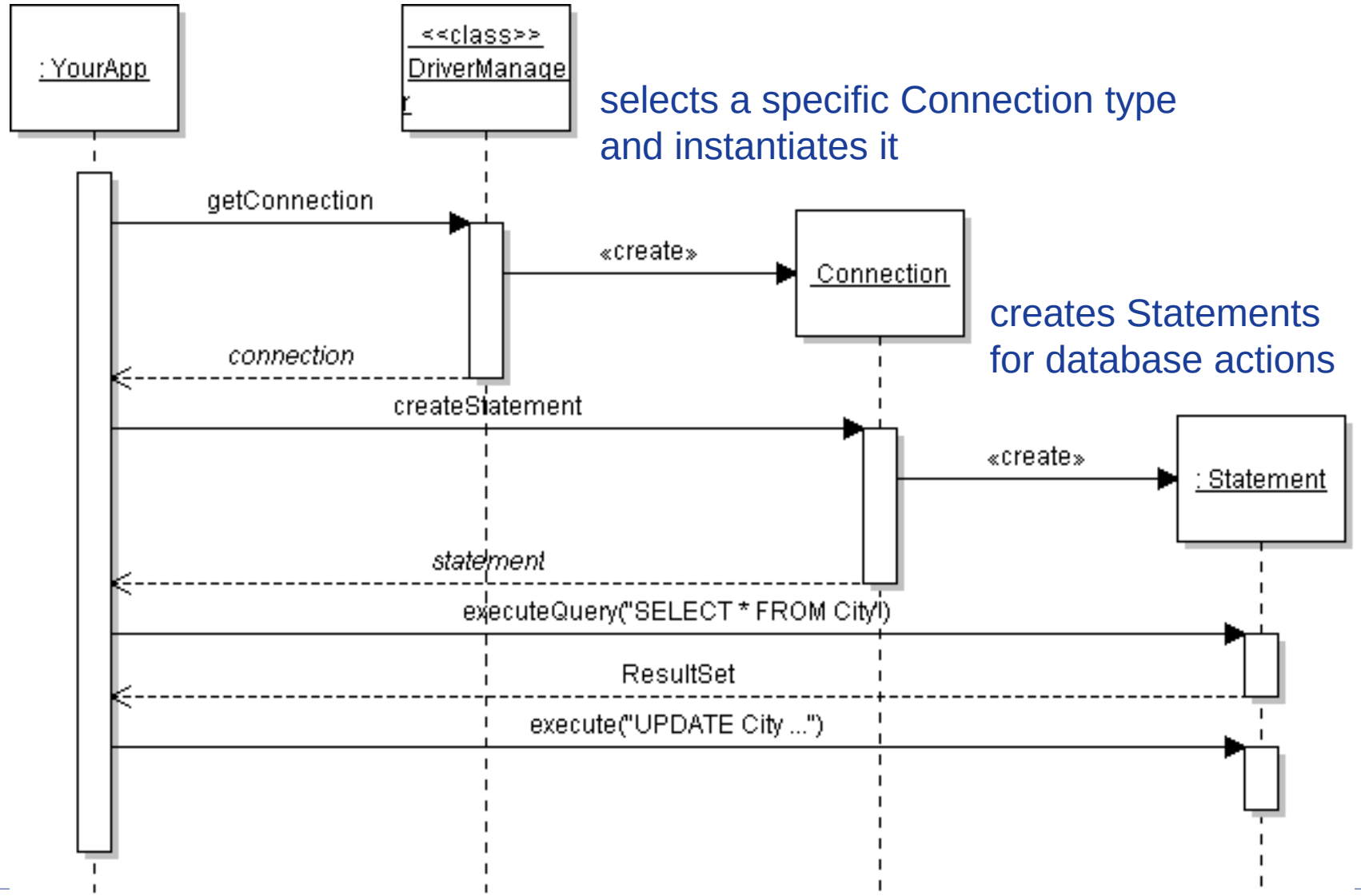
Method	Description
1) public static void registerDriver(Driver driver):	Đăng kí driver với DriverManager.
2) public static void deregisterDriver(Driver driver):	Hủy đăng kí một driver DriverManager.
3) public static Connection getConnection(String url):	Thiết lập kết nối đến CSDL với url cụ thể
4) public static Connection getConnection(String url,String userName,String password):	Thiết lập kết nối đến CSDL với url cụ thể, có username và password

Kết nối CSDL

- Có 5 bước để kết nối các UD Java với CSDL sử dụng JDBC.
  - Đăng kí the Driver class
  - Tạo connection
  - Tạo statement
  - Thi hành truy vấn queries
  - Đóng connection



Các bước làm việc với CSDL



Cách nạp Database Driver?

- Để có thể kết nối trực tiếp đến CSDL, ta cần cài đặt các driver cho ứng dụng.
- Driver được cài đặt trong JAR file.
- Với mỗi loại CSDL khác nhau, ta cần cài đặt phiên bản driver tương ứng cho nó. Ví dụ:

Tên loại Database	Tên driver
Oracle	ojdbc7.jar
MySQL	mysql-connector-java-x.jar
SQL server	sqljdbc4.jar

Nạp driver cho ứng dụng

- JAR phải được khai báo trong classpath:

1. Thêm *jar file* to vào IDE project
2. Thêm JAR file vào CLASSPATH

```
CLASSPATH = /my/path/mysql-connector.jar;.
```

3. Thêm JAR sử dụng Java command line:

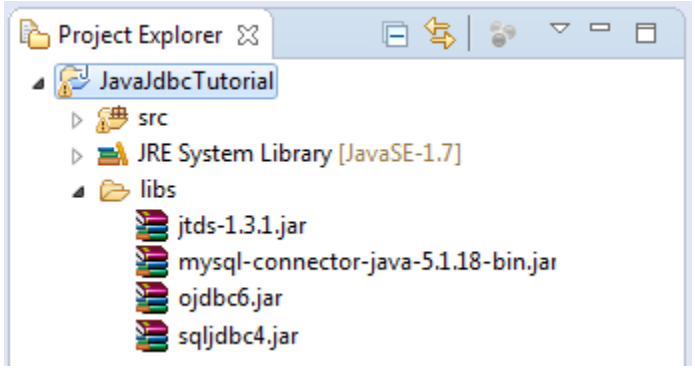
```
java -cp /my/path/mysql-connector.jar ...
```

4. Đặt JAR file vào **`JRE/lib/ext`** directory:

```
C:/java/jre1.6.0/lib/ext/mysql-connector.jar
```

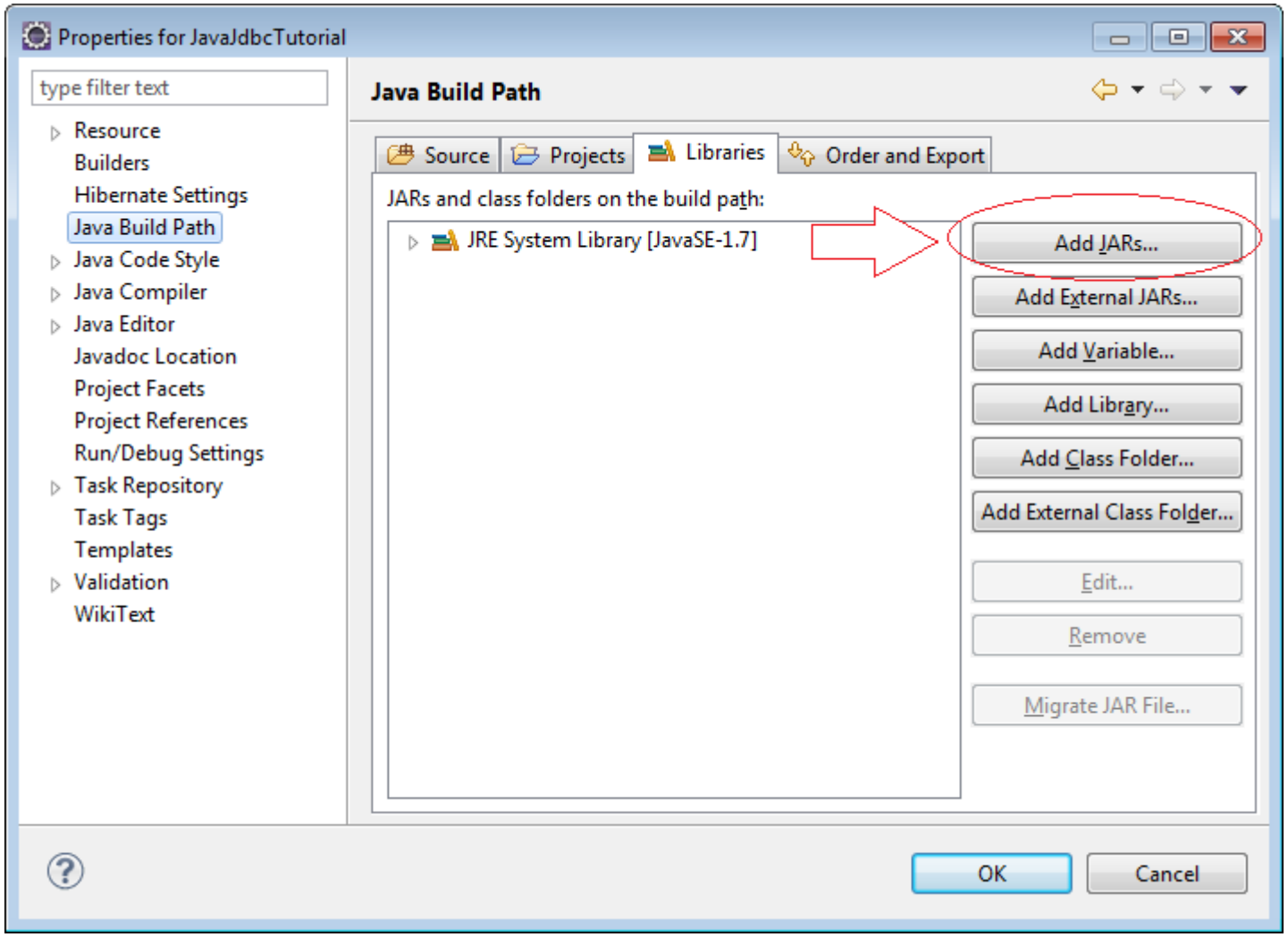
Thêm driver vào build Path

Tạo thư mục libs trên project và copy các thư viện kết nối trực tiếp các loại database *Oracle, MySQL, SQLServer* vào

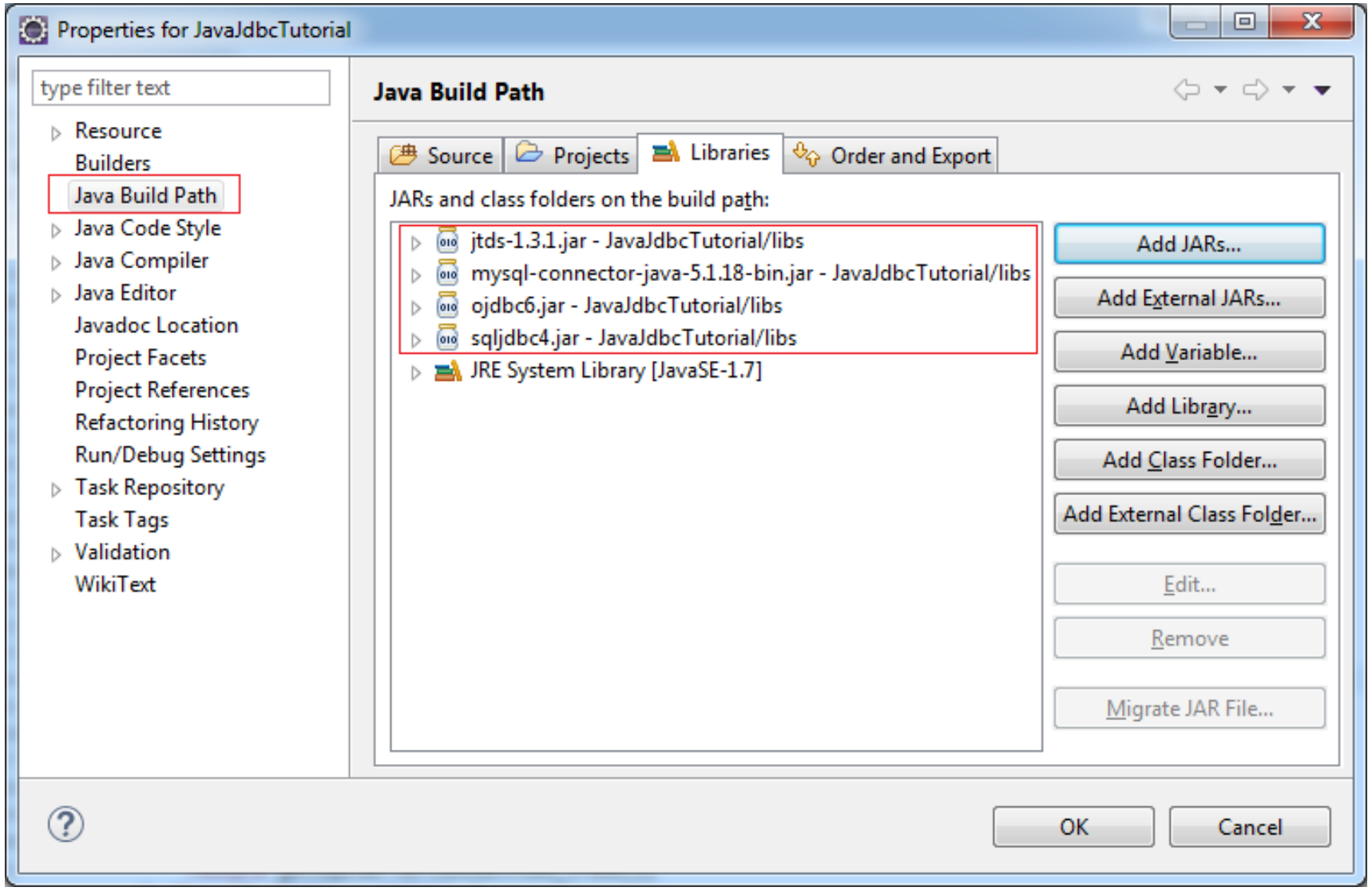


Thêm driver vào build Path

Nhấn phải vào Project chọn Properties:



Thêm driver vào build Path



JDBC Code

```
static final String URL = "jdbc:mysql://dbserver/world";
static final String USER = "student";
static final String PASSWORD = "secret";
// 1.Load driver
// 2. Get a Connection to the database.
Connection connection =
      DriverManager.getConnection( URL, USER, PASSWORD );
// 3. Create a Statement
Statement statement = connection.createStatement();
// 4a. Execute the Statement with SQL command.
ResultSet rs = statement.executeQuery("SELECT * FROM ...");
// 4b. Use the Result.
while ( rs.next( ) ) {
    String name = rs.getString("name");
// 5. Close connection
```

Kết nối CSDL với JDBC

- **java.sql.Connection** là giao diện chuẩn để kết nối đến các DBMS.
- Mỗi DBMS sẽ phải cài đặt interface này.
  - MySQL driver **mysql-connector-java-5.1.7-bin.jar**
- **DriverManager** dùng để chọn driver và thiết lập kết nối.

JDBC URL

- Chỉ định nguồn dữ liệu sẽ kết nối

jdbc:<subprotocol>:<dsn>:<others>

Trong đó:

- ***<subprotocol>***: được dùng để xác định trình điều khiển để kết nối với CSDL.
- ***<dsn>***: địa chỉ CSDL. Cú pháp của ***<dsn>*** phụ thuộc vào từng trình điều khiển cụ thể.
- ***<other>***: các tham số khác

Ví dụ:

- *jdbc:odbc:dbname* là URL để kết nối với CSDL tên dbname sử dụng cầu nối ODBC.
- *jdbc:sqlserver://hostname:1433* là URL để kết nối với CSDL Microsoft SQL Server. Trong đó hostname là tên máy cài SQL Server.

Ví dụ JDBC URL

RDBMS	Database URL format
MySQL	`jdbc:mysql://`*hostname*`:`*portNumber*`/`*databaseName*
ORACLE	`jdbc:oracle:thin:@`*hostname*`:`*portNumber*`:`*databaseName*
DB2	`jdbc:db2:`*hostname*`:`*portNumber*`/`*databaseName*
Java DB/Apache Derby	`jdbc:derby:`*dataBaseName* (embedded) `jdbc:derby://`*hostname*`:`*portNumber*`/`*databaseName* (network)
Microsoft SQL Server	`jdbc:sqlserver://`*hostname*`:`*portNumber*`;databaseName=`*dataBaseName*
Sybase	`jdbc:sybase:Tds:`*hostname*`:`*portNumber*`/`*databaseName*

Database MySQL URL

Định dạng chung của database mysql URL:

```
String DB_URL = "jdbc:mysql://dbserver:3306/world";
```

- Port là TCP port mà hệ QTCSDL sử dụng để lắng nghe yêu cầu.
  - 3306 is the **default port** for MySQL
- Sử dụng **"localhost"** nếu CSDL nằm cùng 1 máy.

Database URL

Hostname và port là tùy chọn.

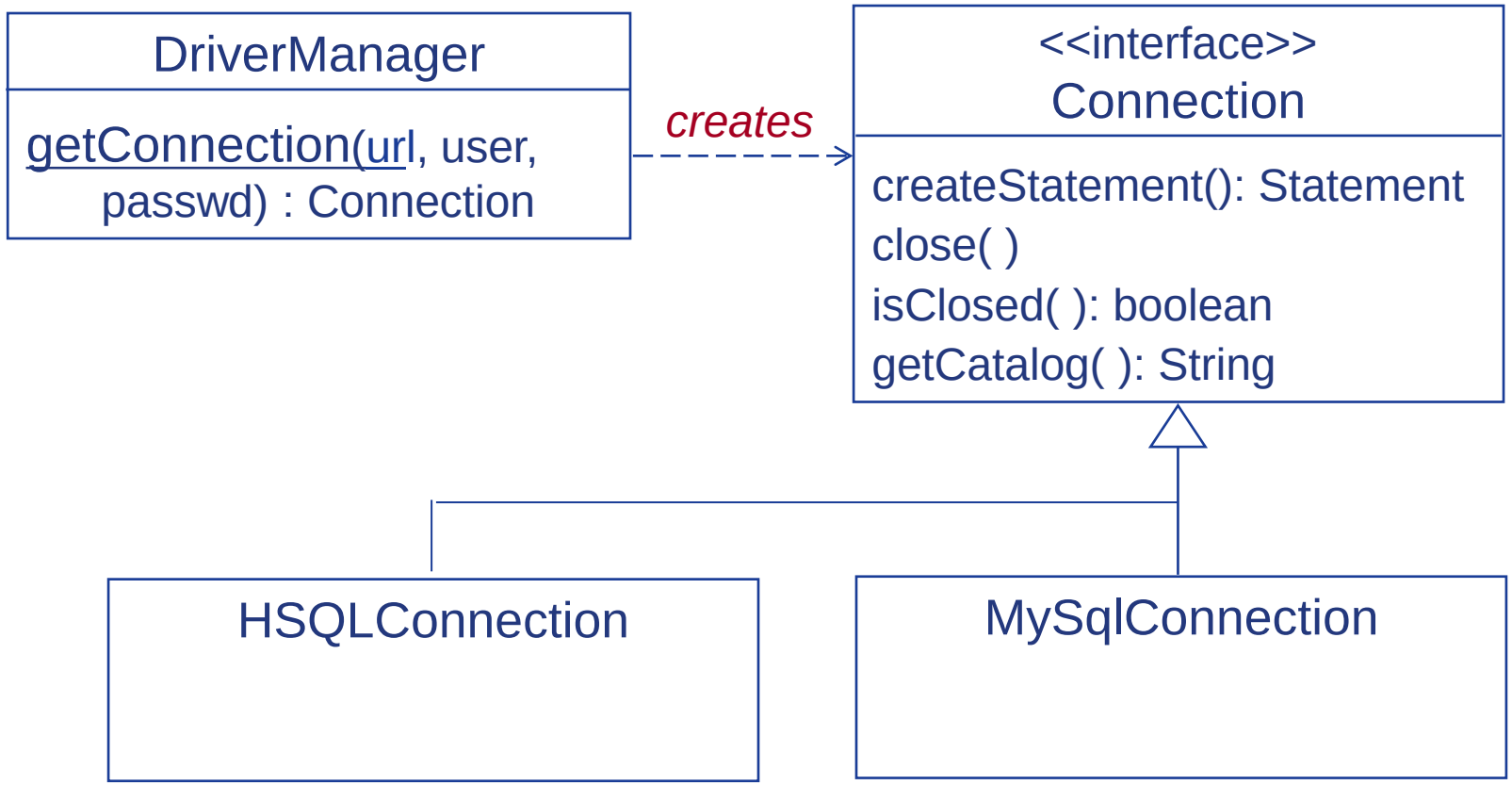
Đối với MySQL driver: defaults là localhost và port 3306

Ví dụ:

```
"jdbc:mysql://localhost:3306/world"
"jdbc:mysql://localhost/world"
"jdbc:mysql:///world"
"jdbc:mysql:/world"
```

Kết nối CSDL với JDBC

url = "jdbc:mysql://hostname/database"



Tạo connection

- Một đối tượng connection là một phiên làm việc giữa ứng dụng java và CSDL.
- Ta có thể tạo ra một đối tượng connection như sau:
- Connection connection =
  DriverManager.getConnection(URL, USER, PASSWORD);

Tóm tắt bài học

- Bài học giới thiệu về JDBC và các kiến thức liên quan đến lập trình CSDL trong Java như các loại drivers, lớp quản lý drivers, các bước kết nối đến cơ sở dữ liệu.
- Sinh viên cần chú ý việc đăng kí các lớp drivers khác nhau tùy vào Hệ quản trị CSDL cụ thể mà mình sử dụng.