Serverske veb tehnologije
- JDBC -

Dragan Ivanović

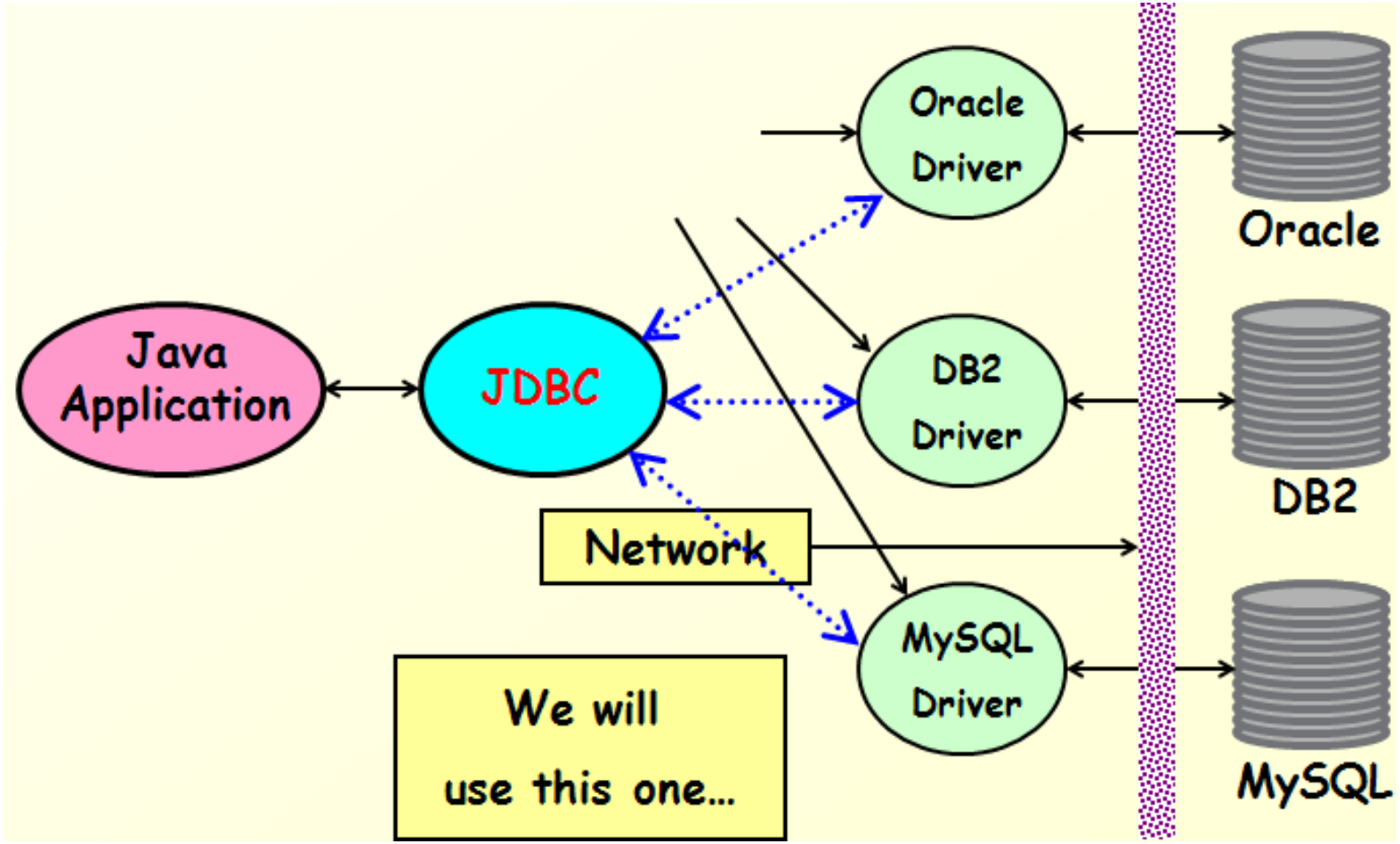
Katedra za informatiku, Fakultet Tehničkih Nauka, Novi Sad

2022.

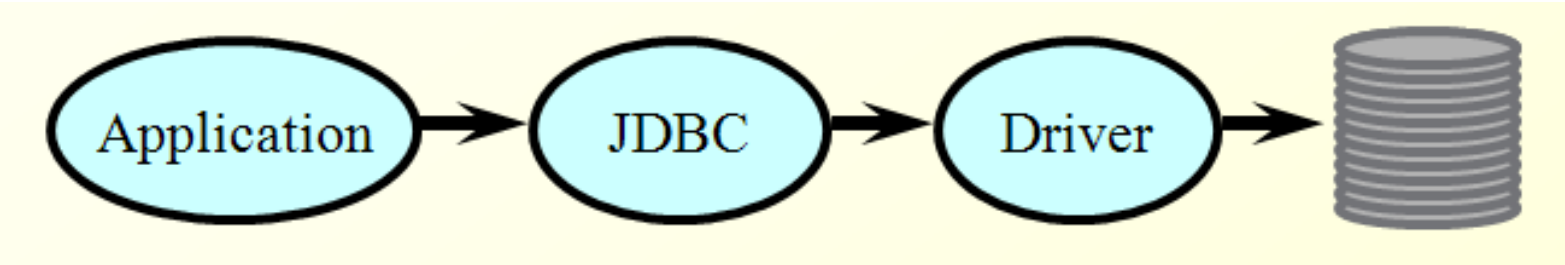
Java Database Connectivity (JDBC)

- Standardan API koji omogućava pristup bilo kom tabularnom izvoru podataka iz Jave
  - relational databases
  - spreadsheets
  - flat files
- Informacije se transformišu iz relacionog modela u objektni model i obrnuto
  - relacioni model je dobar za skladištenje podataka
  - objektni model je dobar za programiranje

Arhitektura



Arhitektura



Način rada

- Učitavanje jednog ili više *driver*-a
- Specificiraju se parametri konekcije
- *Driver* komunicira sa bazom podataka
- Aplikacija izdaje naradbe koje *driver* izvršava nad bazom podataka
- Cilj: može se promeniti SUBP bez promene koda

JDBC driver za MySQL

- Skinuti binarnu distribuciju sa `http://dev.mysql.com/downloads/connector/j/5.1.html`
- Otpakovati arhivu i staviti mysql-connector-java-[version]-bin.jar u classpath projekta (mysql-connector-java-5.1.5-bin.jar)
- Online dokumentacija je dostupna ovde: `http://dev.mysql.com/doc/connector-j/en/index.html`

Koraci u radu

- Učitavanje *driver*-a
- Specificiranje parametara konekcije i preuzimanje konekcije
- Kreiranje *Statement* objekta
- Izvršenje upita ili update-a upotrebom *Statement* objekta
- Procesiranje rezultata
- Zatvaranje konekcije

Učitavanje *driver*-a

- *Driver* se može učitati indirektno naredbom:
- `Class.forName("com.mysql.jdbc.Driver");`
- *Class.forName* učitava klasu putem *class loader*-a
- Kada je *mysqlDriver* učitan, automatski se
  - kreira jedna instanca ove klase
  - registruje ova instanca u *DriverManager*-u

Specificiranje parametara konekcije i preuzimanje konekcije

- Baza podataka je identifikovana URL-om
- *DriverManager* traži driver među registrovanim koji odgovara bazi podataka datog URL-a

```
Connection conn = DriverManager.getConnection(
        "jdbc\:mysql\://localhost\:3306/osa", "osa", "osa");
```

Interakcija sa bazom podataka

- Koristimo Statement objekat da bi:
  - pretraživali podatke u bazi podataka (query)
  - menjali podatke u bazi podataka (update)
- Tri vrste interfejsa: *Statement*, *PreparedStatement*, *CallableStatement*
- Implementacije ovih interfejsa su u implementaciji *driver*-a

Pretraga podataka u bazi podataka

```
String queryStr =
"SELECT * FROM employee " +
"WHERE lname = 'Wong'";

Statement stmt = con.createStatement();
ResultSet rs = stmt.executeQuery(queryStr);
```

- *ResultSet* - rezultati pretrage

Izmena podataka u bazi podataka

```
String deleteStr =
"DELETE FROM employee " +
"WHERE lname = 'Wong'";

Statement stmt = con.createStatement();
int delnum = stmt.executeUpdate(deleteStr);
```

- *executeUpdate* - povratna vrednost je broj redova koji su pretrpeli izmene u bazi podataka

PreparedStatement

- Za upite (ili izmene) koji se izvršavaju više puta
- Parsiraju se i kompajliraju od strane SUBP-a samo jednom
- Parametrizovane vrednosti se postavljaju nakon kompajliranja umesto ?

Pretraga podataka u bazi podataka

```
String queryStr =
"SELECT * FROM employee " +
"WHERE superssn= ? and salary > ?";

PreparedStatement pstmt =  con.prepareStatement(queryStr);

pstmt.setString(1, "333445555");
pstmt.setInt(2, 26000);

ResultSet rs = pstmt.executeQuery();
```

Izmena podataka u bazi podataka

```
String deleteStr =
"DELETE FROM employee " +
"WHERE superssn = ? and salary > ?";

PreparedStatement pstmt =  con.prepareStatement(deleteStr);

pstmt.setString(1, "333445555");
pstmt.setDouble(2, 26000);

int delnum = pstmt.executeUpdate();
```

Da li je ovo isto?

- ```
  String val = "abc";
  PreparedStatement pstmt =
  con.prepareStatement("select * from R where A=?");
  pstmt.setString(1, val);
  ResultSet rs =  pstmt.executeQuery();
  ```
- ```
  String val = "abc";
  Statement stmt =  con.createStatement( );
  ResultSet rs =
  stmt.executeQuery("select * from R where A=" + val);
  ```

CallableStatement

- Za izvršavanje uskladištenih procedura
- Brža komunikacija sa bazom podataka

ResultSet

- *ResultSet* objekti omogućavaju pristup tabelama koji su rezultat izvršavanja upita
- Samo jedan *ResultSet* po *Statement* objektu može biti otvoren u jednom trenutku
- Sekvenca redova tabele, pomoću metode next() pristupamo narednom redu.

ResultSet - metode

- *boolean next()* - aktivira sledeći red, prvi put kada se pozove pozicioniramo se na prvi red, vraća false kada nema više redova u rezultatu
- *Type getType(int columnNumber)*
  - *getString(5)*, *getInt(2)*, prvi u nizu je 1, ne 0
- *Type getType(String columnName)*
  - *getString("name")*, *getInt("id")*
- *int findColumn(String columnName)*
- *void close()*

Primer *ResultSet*

```
Statement stmt = con.createStatement();
ResultSet rs =
stmt.executeQuery("select lname, salary from Employees");
//Print the result
while(rs.next()){
System.out.print(rs.getString(1) + ":");
System.out.println(rs.getDouble("salary"));
}
```

SQL i Java tipovi

SQL type	Java Type
CHAR, VARCHAR, LONGVARCHAR	String
NUMERIC, DECIMAL	java.math.BigDecimal
BIT	boolean
TINYINT	byte
SMALLINT	short
INTEGER	int
BIGINT	long
REAL	float
FLOAT, DOUBLE	double
BINARY, VARBINARY, LONGVARBINARY	byte[]
DATE	java.sql.Date
TIME	java.sql.Time
TIMESTAMP	java.sql.Timestamp

Null vrednosti

- *NULL* u *SQL*-u znači da je kolona prazna, nije isto kao 0 ili ""
- *ResultSet.wasNull(columnNumber)*
- *ResultSet.getInt(columnNumber)* ne razlikuje 0 i *NULL*
- *PreparedStatement.setNull(index, Types.sqlType)* - za primitivne tipove
- *PreparedStatement.setType(index, null)* - za objekte

Važne napomene

- zatvaranje objekata: Connection, Statement, ResultSet upotrebom metode *void close()* je poželjno
- obrada izuzetaka je poželjna
- rad sa transakcijama kada je to potrebno

Upotreba

- Apache DbUtils (`http://jakarta.apache.org/commons/dbutils/`)
- ORM (Object Relational Mappers):
  - Hibernate (`http://www.hibernate.org/`),
  - JDO (`http://java.sun.com/products/jdo/`),
  - TopLink (`http://www.oracle.com/technology/products/ias/toplink/index.html`)
- Spring i JDBC - https://docs.spring.io/spring/docs/current/spring-framework-reference/html/jdbc.html

Spring boot Jdbc

- Spring Primer 15