

# Innehåll

Hur ansluter jag min java applikation till databasen?

- Vad är JDBC? Var måste databasen finnas?
- Hur ser arkitekturen ut?
- Nyckeldelar

## JDBC

JDBC - Java Databasbase connectivity

Standard API för informations kommunikation mot en relationsdatabas. JDBC stödjer SQL, dvs. du är begränsad till vad du kan om SQL i kommunikation från din java applikation till databasen. JDBC fungerar lokalt, över nätverk och molnet.

Stödjer bla. MySQL, Oracle, SQL Server

JDBC Aritektur

Applikation ----> JDBC API ----> JDBC Driver Manager ----> Driver ----> Relationsdatabas

Nyckeldelar

- DriverManager - ser till så vi tilldelas rätt driver
- Connection - Brygger en anslutning till databasen samt håller den. Innehåller även information om databasens struktur
- Statement - Tillåter oss att köra SQL frågor, samt hämta ner resultatet
- Resultset - Kan användas för att uppdatera data, samt att visa datan som en abstrakt representation av tabellen.

JDBC url defineras som

- jdbc:<protokol>:anslutningsData

ex. jdbc:mysql://localhost:3306/demo är en *mysql* databas med namnet *demo* som vi ansluter mot via *jdbc drivern* lokalt på *port 3306*.

## Mer om resultset

ResultSet kan vara av tre typer:

Vilken man behöver beror på vad man vill göra med data man erhåller:

Typ 1: Om man bara vill gå igenom data från början till slut. TYPE\_FORWARD\_ONLY

Typ2: Om man vill gå igenom data framåt men OCKSÅ bakåt och samtidigt upptäcka om det gjorts ändringar. TYPE\_SCROLL\_SENSITIVE

Typ3: Om man vill gå igenom data framåt men OCKSÅ bakåt men bryr sig inte att upptäcka om det gjorts ändringar. TYPE\_SCROLL\_INSENSITIVE

Vilken typ av ResultSet vi erhåller anger vi som parameter när vi skapar upp vårt Statement-objekt. Om vi väljer typ 2 eller 3 måste vi även ange om data ska kunna manipuleras eller ej. Detta görs med CONCUR\_READ\_ONLY respektive CONCUR\_UPDATEABLE som en andra parameter till Statement-objektet.

Holdability avser när resultseten släpps/stängs, mer om commit vid senare tillfälle.

Ett ResultSet har en pekare (cursor) för att hålla reda på vilken rad man befinner sig på. När man erhåller sitt ResultSet står den och pekar precis innan första raden.

Allt detta sammanfattas som

- ResultSetType - TYPE\_SCROLL\_SENSITIVE, TYPE\_SCROLL\_INSENSITIVE, TYPE\_FORWARD\_ONLY
- ResultSetConcurrency - CONCUR\_READ\_ONLY, CONCUR\_UPDATEABLE
- ResultSetHoldability - HOLD\_CURSORS\_OVER\_COMMIT, CLOSE\_CURSORS\_AT\_COMMIT

Vanliga metoder för att traversera data är :

- next()-flyttar pekaren ett steg framåt. Returnerar true om det finns en rad, false om vi nåt slutet.
- previous()-flyttar bakåt, Returnerar true om det finns en rad, false om vi nåt början.
- first()-flyttar pekaren först
- last()-flyttar pekaren sist
- absolute(int n)-flyttar till en specifik rad
- isBeforeFirst()- kollar om pekaren står före första raden
- isFirst()-kollar om pekaren står på första raden
- isLast()-Kollar om pekaren står på sista raden
- isAfterLast()-Kollar om pekaren står efter sista raden
- getString(String columnName) för att hämta värdet från en column på en rad givet namnet på kolumnen
- getInt(int columnRow) för att hämta värdet från en column på en rad givet indexet på kolumnen (1-indexerat)

## Sample Code

```
import java.sql.*;
```

```
....
```

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

```
    //Ifall JDBC Driver version > 4
```

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

```
    //1. Hämta anslutning, här är användarnamnet "root" och lösenordet tomt, dvs. ""
```

```
    Connection conn = DriverManager.getConnection("jdbc:mysql://localhost/dbName",  
    "root", "");
```

```
    //2. Skapa ett statment, här sätter vi inga "resultset typer"
```

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

```
    //3. Hämta resultatet
```

```
    ResultSet rs = stmt.executeQuery("SELECT * FROM people");
```

```
    //4. Processera resultatet
```

```
    while(rs.next()){
```

```
        System.out.println(rs.getString("name");
```

```
    }
```

```
}
```