MySQL Database aanspreken vanuit IntelliJ

1 USB WebServer

De snelste manier om een MySql-database op te zetten, is gebruik maken van USB Webserver.

1.1 Benodigdheden

Software	Versie
IntelliJ	Versie 2018
Java Development Kit (JDK)	Versie 8
USB Webserver	Versie 8.6

1.2 Downloaden van USB Webserver

Navigeer naar de download pagina van USB Webserver op http://www.usbwebserver.net/webserver/ en download USB Webserver. Sla het bestand op in een tijdelijke map.

1.3 Installeren van USB Webserver

Pak het zojuist afgehaalde bestand uit in een zelfgekozen map. USB Webserver zelf vereist geen installatie. Om USB Webserver te gebruiken moet je slechts klikken op Usb Webserver.exe

1.4 Rondleiding van USB Webserver

Hoofdscherm

Op het hoofdscherm van Usb Webserver vind je rechts enkele tabbladen (algemeen, Apache, MySQL, Instellingen en Over). Je kan er ook meteen de status van MySQL en Apache aflezen.

MySQL en Apache moeten actief ☑ zijn.

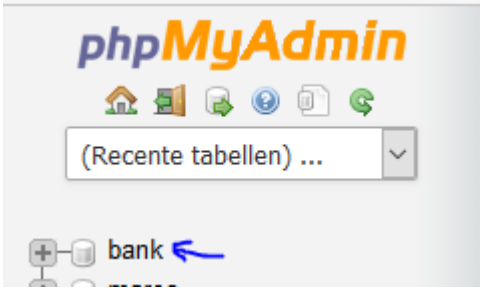
Wanneer Apache niet draait, kan dit omdat de poort 8080 al bezet is. Klik op Instellingen en verander de poort van Apache naar bv. 8484

Onder tabblad Algemeen staan een aantal handige snelkoppelingen. Via de snelkoppeling Localhost navigeer je naar de homepage van je lokale server. Om de database te beheren maak je gebruik van PHPMyAdmin (login: root, paswoord: usbw).

1.5 Database maken met USB Webserver

1. Start PHPMyAdmin op
2. Kies voor tabblad Databanken en maak een nieuwe databank 'bank' aan.

3. Selecteer de databank 'bank'



4. Kies voor tabblad SQL en kopieer het script en klik op start om het script uit te voeren.

5. Na het uitvoeren is de database gecreëerd. Nu kunnen we de nodige stappen ondernemen in Netbeans.

2 IntelliJ

2.1 MySQL-driver

Er is een mySQL-driver nodig om de database aan te spreken. In je POM-file is hiervoor een 'dependency' aangemaakt:

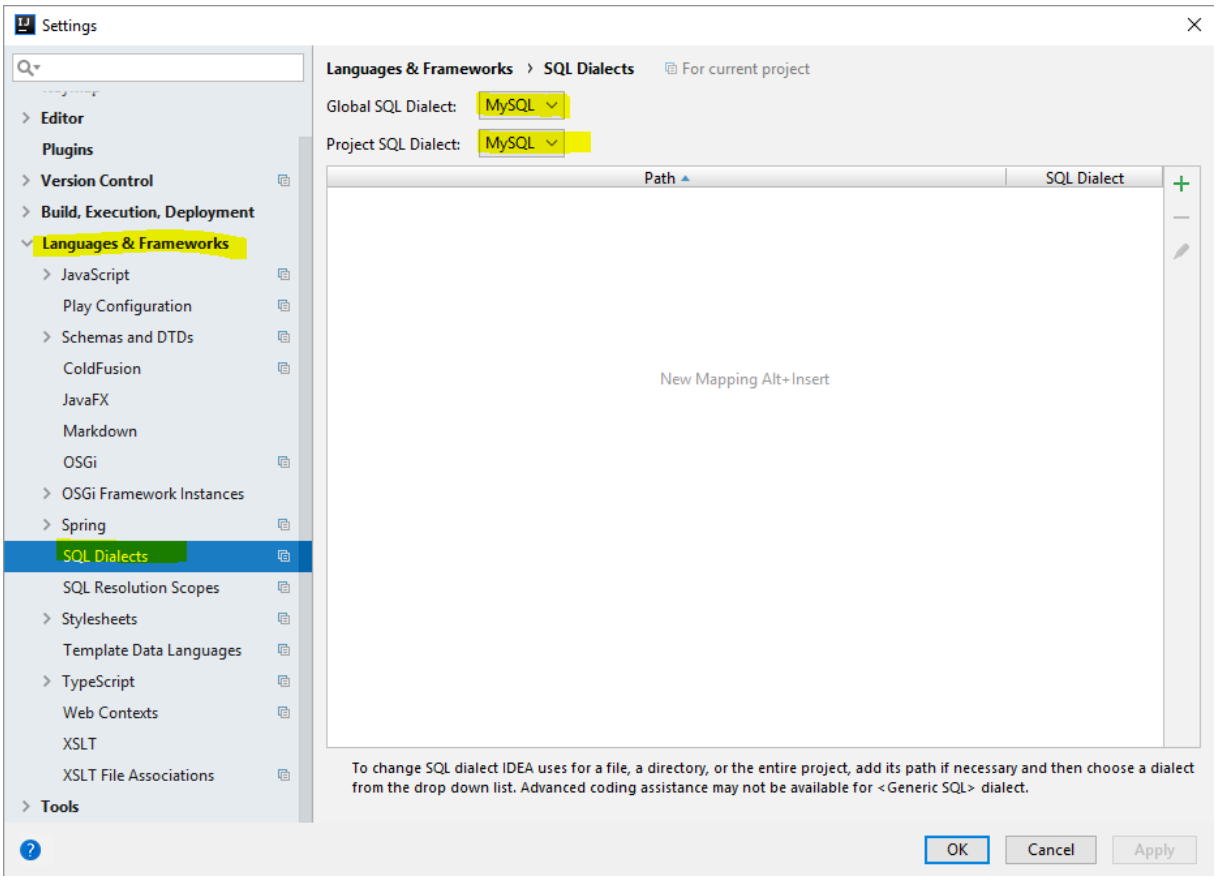
```
<dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
    <version>8.0.16</version>
</dependency>
```

2.2 Databasetoegang vanuit IntelliJ

Voordeel: SQL-code wordt herkend in je project

Projectinstellingen

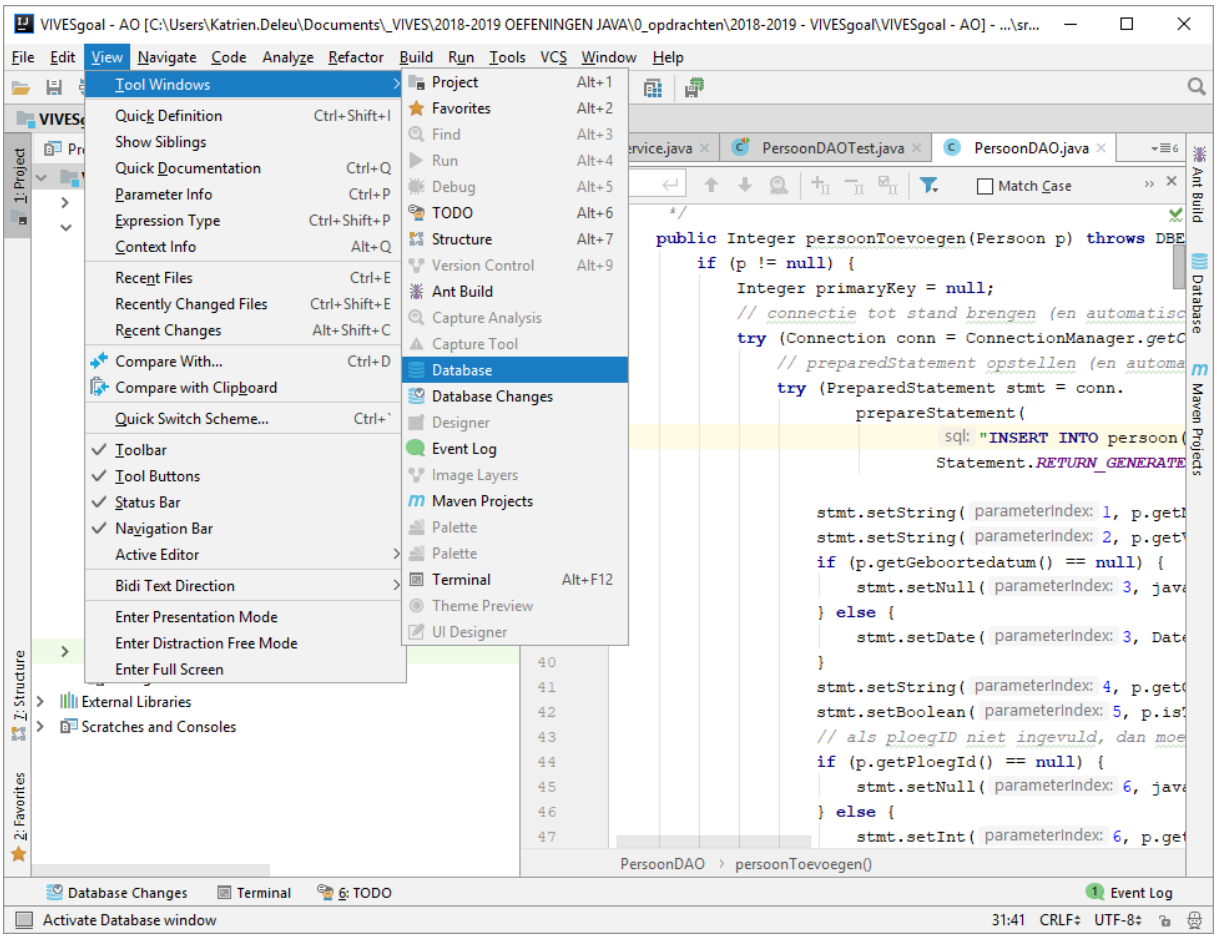
Bij de projectinstellingen (CTRL-ALT-S) kun je het gebruikte dialect opgeven (onder Languages & Frameworks > SQl Dialects). Zet Gloabel SQL Dialect en Project SQL Dialect op MySQL. Dit zorgt ervoor dat er gecontroleerd wordt of er geen syntaxfouten zitten in je query.



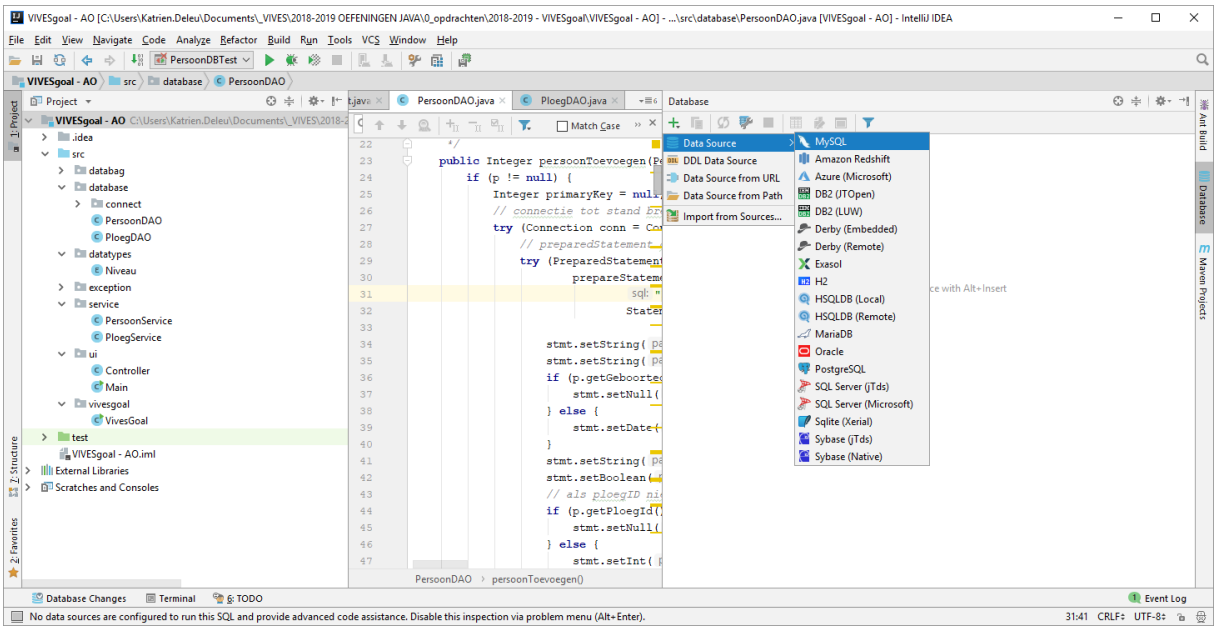
Databasevenster en connectie

Verder kun je ook de connectie maken met de database. Op die manier worden de SQL-queries ook gecontroleerd op kolomnamen ed.

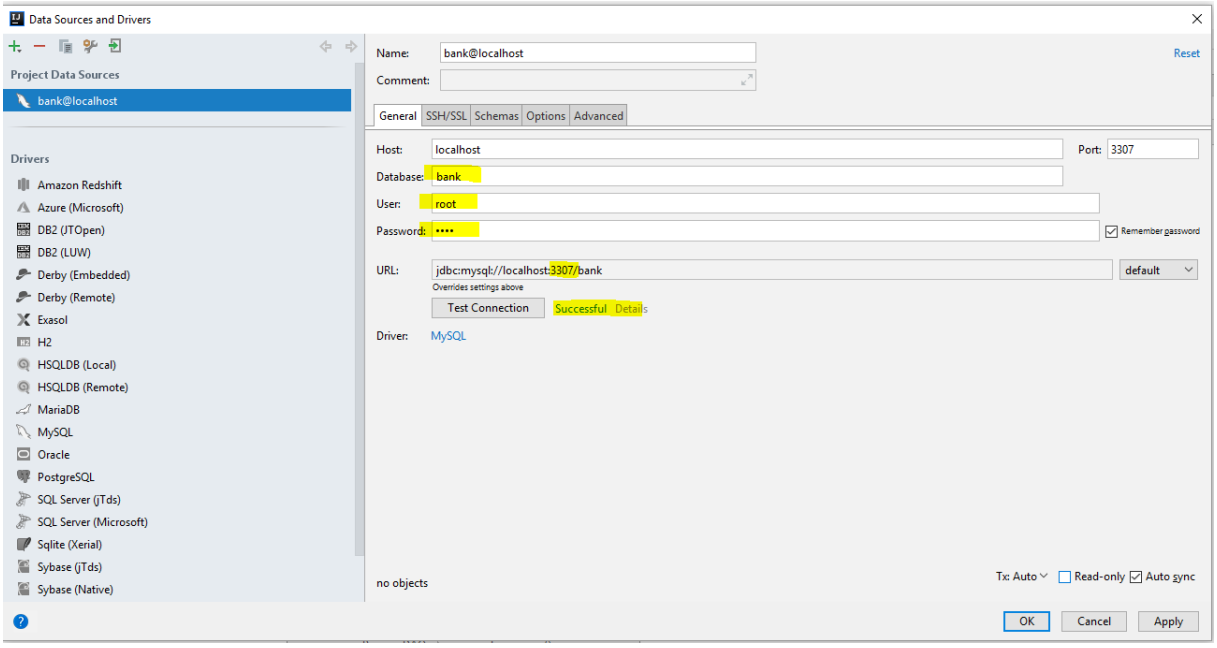
Kies View, Tool windows en klik Database aan.



Klik op + en kies voor data source, MySQL



Vul alle gegevens in en zorg dat je USBServer draait. Je kan de connectie testen met ‘Test Connection’



2.3 De eerste regels code

Als je zelf een nieuw project schijft, dan is het verstandig eerst te testen of de connectie goed werkt. Dit kan je doen door een eenvoudige main-klasse te schrijven die gebruik maakt van de ConnectionManager-klasse, de DBProp-klasse en de DatabaseException-klasse uit de “eenvoudige bank”-oefening gecombineerd met een properties-file met alle gegevens in om de connectie te kunnen leggen. Zorg dat de klassen ConnectionManager en DB-prop in de package ‘database.connect’ zitten. De Exception-klasse steek je in de package ‘exception’. De properties-file zet je onder de projectmap.

Met deze klassen kan de connectie met de database getest worden aan de hand van een main()-methode met code zoals hieronder aangegeven. Zorg dat er data in de tabellen zit!

```
import database.ConnectionManager;
import exception.DBException;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;

public class TEST_DB {
  public static void main(String args[]) {
    try (Connection conn = ConnectionManager.getConnection();) {
      // preparedStatement opstellen (en automtisch sluiten)
      try (PreparedStatement stmt = conn.prepareStatement(
                                          "select * from Klant");) {
        // execute voert het SQL-statement uit
        stmt.execute();
        // result opvragen (en automatisch sluiten)
        try (ResultSet r = stmt.getResultSet()) {
          while (r.next()) {
```

```
            System.out.println(r.getString("voornaam") + " " +
                               r.getString("naam"));
          }
        } catch (SQLException sqlEx) {
          System.out.println("SQL-exception - resultset");
        }
      } catch (SQLException sqlEx) {
        System.out.println("SQL-exception - statement");
      }
    } catch (SQLException sqlEx) {
      System.out.println("SQL-exception - connection");
    } catch (DBException db) {
      System.out.println(db.getMessage());
    }
  }
}
```