# H2 related configurations

<dependency>  
 <groupId>com.h2database</groupId>  
 <artifactId>h2</artifactId>  
 <version>2.2.220</version>  
 <scope>**compile**</scope>  
</dependency>

1. Make sure this dependency is since compile time onwards

Only when this jar is and data jpa jar is also mandatory from which H2Dialect will comes

1. So ensure data jpa jar is also present

<dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-data-jpa</artifactId>  
</dependency>

For sql server url is

spring.datasource.url=jdbc:sqlserver://localhost:14330;database=pub;user=sa;password=aa

jdbc:h2://localhost:8080:testdb

jdbc:mysql://localhost:3306;database=mani;user=root;password=MANIdeep@1234

jdbc:mysql://localhost:3306/mani;user=root;password=MANIdeep@1234

3) add all these properties

#for h2 these are all the properties  
spring.datasource.url=jdbc:h2:mem:testdb  
spring.datasource.driverClassName=org.h2.Driver  
spring.datasource.username=sa  
spring.datasource.password=1234  
spring.jpa.database-platform=org.hibernate.dialect.H2Dialect  
#unless u enable, h2 cant be accessed from browser

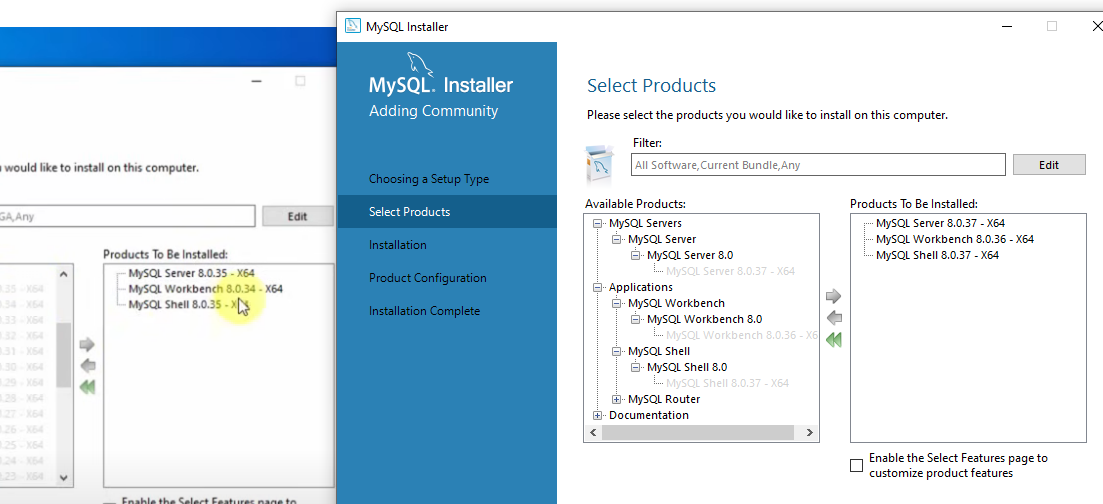
spring.h2.console.enabled=true

spring.jpa.show-sql=true

1. Launch h2 console at following place

<http://localhost:8080/h2-console>

# Install my sql



While installing choose setup type as custom and select all 3 required softwares

# MYSQL related properties

When my sql is not started goto services.msc🡪 start MYSQL80

spring.datasource.url=jdbc:mysql://localhost:3306/mani?createDatabaseIfNotExist=true&useUnicode=true  
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver  
spring.datasource.username=manideep  
spring.datasource.password=MANIDEEP  
spring.jpa.show-sql=true  
spring.jpa.hibernate.ddl-auto=update  
spring.jpa.properties.hibernate.dialect = org.hibernate.dialect.MySQL8Dialect  
spring.jpa.databse-platform = org.hibernate.dialect.MySQL8Dialect

## Custom property names – creating 2 datasources

|  |  |
| --- | --- |
| creating DataSource using spring boot annotations | Manual bean creation (here we should directly use hikari datasource) |
| spring.old-datasource.jdbc-url=jdbc:mysql://localhost:3306/sprbatch spring.old-datasource.username=mani spring.old-datasource.password=mani spring.old-datasource.driverClassName=com.mysql.cj.jdbc.Driver  spring.new-datasource.jdbc-url=jdbc:mysql://localhost:3306/aims spring.new-datasource.username=santu1 spring.new-datasource.password=santu1  @Bean("primaryDataSource") @Primary @ConfigurationProperties("spring.old-datasource") public *DataSource* createMainDataSource(){  System.***out***.println("creating data source");  *DataSource* build = DataSourceBuilder.*create*().build();  return build; } @Bean(value = "secondaryDataSource") @ConfigurationProperties("spring.new-datasource") public *DataSource* secondaryDataSource(){  System.***out***.println("creating secondary data source");  *DataSource* build = DataSourceBuilder.*create*().build();  return build; } | public *DataSource* createMainDataSource(){  HikariDataSource hds=new HikariDataSource();  hds.setJdbcUrl("jdbc:mysql://localhost:3306/sprbatch");  hds.setUsername("mani");  hds.setPassword("mani");  return hds; } |
|  |  |

spring.old-datasource.jdbc-url=jdbc:mysql://localhost:3306/sprbatch

## in the above we should not use key called “url “, we should use only key called “jdbc-url”

As these keys will be binded to hikari datasource, in hikari datasource we have only “jdbcUrl” instead of “url” and spring boot by default

uses this datasource  
spring.old-datasource.username=mani  
spring.old-datasource.password=mani  
spring.old-datasource.driverClassName=com.mysql.cj.jdbc.Driver

1. If u are defining custom keys, we should tell how to bind
2. and manually we should create datasource as below

@Bean  
@Primary  
@ConfigurationProperties("spring.old-datasource")  
public *DataSource* createMainDataSource(){  
 System.***out***.println("creating data source");  
 *DataSource* build = DataSourceBuilder.*create*().build();  
 System.***out***.println("Data source building is success" );  
 return build;  
}

# These cant be used

app.datasource.url=jdbc:mysql://localhost/test

app.datasource.username=dbuser

app.datasource.password=dbpass

app.datasource.pool-size=30

However, there is a catch. Because the actual type of the connection pool is not exposed, no keys are generated in the metadata for your custom DataSource and no completion is available in your IDE (because the DataSource interface exposes no properties). Also, if you happen to have Hikari on the classpath, this basic setup does not work, because Hikari has no url property (but does have a jdbcUrl property). In that case, you must rewrite your configuration as follows:

app.datasource.jdbc-url=jdbc:mysql://localhost/test

app.datasource.username=dbuser

app.datasource.password=dbpass

app.datasource.maximum-pool-size=30

public class HikariConfig {

private String jdbcUrl;

}

1. fdg

# Sample queries

U can login to command prompt and type below commands

Here by default user name will be “root”

|  |  |
| --- | --- |
| To create database | “create database <ur db name>” it will create database |
| To see database list | “show databases” |
| To create user 🡪  Note:- only root user can create databases/schemas  Normal user cant create and see those db & tables unless it was granted for him | CREATE USER 'mani'@'localhost' IDENTIFIED BY 'mani' |
| Grant view or modify access to certain user  Note:- here db name “sprbatch” is created by root user, hence by default other users cant see those db, Hence we are giving all privileges to user named “mani” for a db/schema named “sprbatch” only then mani can see that database  Until u give that privilege mani cant see that database at all | GRANT ALL PRIVILEGES ON sprbatch.\* TO 'mani'@'localhost' |
| To see all privileges | SHOW GRANTS FOR 'mani'@'localhost'; |
| To delete a database | drop database aims2 |

<https://www.tutorialspoint.com/h2_database/h2_database_select.htm>

**Create query**

CREATE TABLE student (

id INT NOT NULL,

name VARCHAR(50) NOT NULL,

address VARCHAR(20) NOT NULL

);

## Mysql properties

spring.datasource.url=jdbc:mysql://localhost:3306/mani?createDatabaseIfNotExist=true&useUnicode=true  
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver  
spring.datasource.username=manideep  
spring.datasource.password=MANIDEEP  
spring.jpa.show-sql=true  
spring.jpa.hibernate.ddl-auto=create  
spring.jpa.properties.hibernate.dialect = org.hibernate.dialect.MySQL8Dialect  
spring.jpa.databse-platform = org.hibernate.dialect.MySQL8Dialect