(00:16:18)

Для успешного создания бина DataSource необходимо в application.yml прописать следующее:

spring:  
 datasource:  
 url: jdbc:h2:mem:test  
 username: sa  
 password:  
 driver-class-name: org.h2.Driver

Кусок кода, внесенный в main() (временно) для демонстрации механизма работы с данными:

ConfigurableApplicationContext context = SpringApplication.*run*(SpringDataDemoApplication.class, args);  
  
DataSource dataSource = context.getBean(DataSource.class);  
  
try (Connection connection = dataSource.getConnection()) {  
 try (Statement statement = connection.createStatement()) {  
 statement.execute("create table users(id bigint, name varchar(512))");  
 }  
  
 try (Statement statement = connection.createStatement()) {  
 statement.execute("insert into users(id, name) values(1, 'Igor')");  
 }  
  
 try (Statement statement = connection.createStatement()) {  
 ResultSet resultSet = statement.executeQuery("select id, name from users");  
 while (resultSet.next()) {  
 System.*out*.println(resultSet.getInt("id"));  
 System.*out*.println(resultSet.getString("name"));  
 }  
 }  
}  
 }

(00:28:36)

**Бин JdbcTemplate**

ConfigurableApplicationContext context = SpringApplication.*run*(SpringDataDemoApplication.class, args);  
JdbcTemplate jdbcTemplate = context.getBean(JdbcTemplate.class);  
  
//JdbcTemplate  
  
jdbcTemplate.execute("create table users(id bigint, name varchar(512))");  
jdbcTemplate.execute("insert into users(id, name) values(1, 'Igor')");  
  
List<User> users = jdbcTemplate.query("select id, name from users", new RowMapper<User>() {  
 @Override  
 public User mapRow(ResultSet rs, int rowNum) throws SQLException {  
 return new User(rs.getLong("id"), rs.getString("name"));  
 }  
});

*/\*\*  
 \* spring-data-jdbc - набор готовых инструментов для взаимодействия с базой данных.  
 \* По сути, оборачивает стандартный JDBC и предоставляет удобные интерфейсы для настройки и   
 \* взаимодействия с БД.  
 \*/*

(00:41:02)

*/\*\*  
\*JDBC JPA  
 \*  
 \* JDBC - библиотека внутри Java для работы с базами данных  
 \*Driver, Connection, Statement - основные интерфейсы JDBC  
 \*  
 \* JPA (Jakarta Persistence API)- набор соглашений по работе с реляционными моделями.  
 \* Основнаяя идея - "замапить" DB-модель и работать со строками таблиц как с объектами.  
 \*JPA - это не реализация, а протокол (api, спецификация).  
 \* Hibernate - одна из реализаций JPA.  
 \*  
 \* spring-data-jdbc - набор готовых инструментов для взаимодействия с базой данных.  
 \* По сути, оборачивает стандартный JDBC и представляет удобные интерфейсы для настройки и взаимодействия с БД.  
 \*  
 \* spring-data-jpa - набор готовых инструментов для работы с JPA.  
 \*  
\*\*/*

Конвенция JPA query methods (правила именования методов) здесь:

https://docs.spring.io/spring-data/jpa/reference/repositories/query-keywords-reference.html