

Search...

[Advance Java Course](#) [Java Tutorial](#) [Java Spring](#) [Spring Interview Questions](#) [Java SpringBoot](#) [Spring](#)

# Spring Data JPA - Attributes of @Column Annotation with Example

Last Updated : 18 Mar, 2025

**Spring Data JPA** is a powerful framework that simplifies database interactions in Spring Boot applications. The **@Column annotation in Spring Data JPA** is widely used to customize column properties such as length, default values, and constraints in a database table. Understanding how to use @Column effectively helps optimize database performance and maintain data integrity.

In this article, we will explore the most important **attributes of the @Column annotation in Spring Boot applications** with practical examples.

## Prerequisites:

Before we start, ensure you have the following:

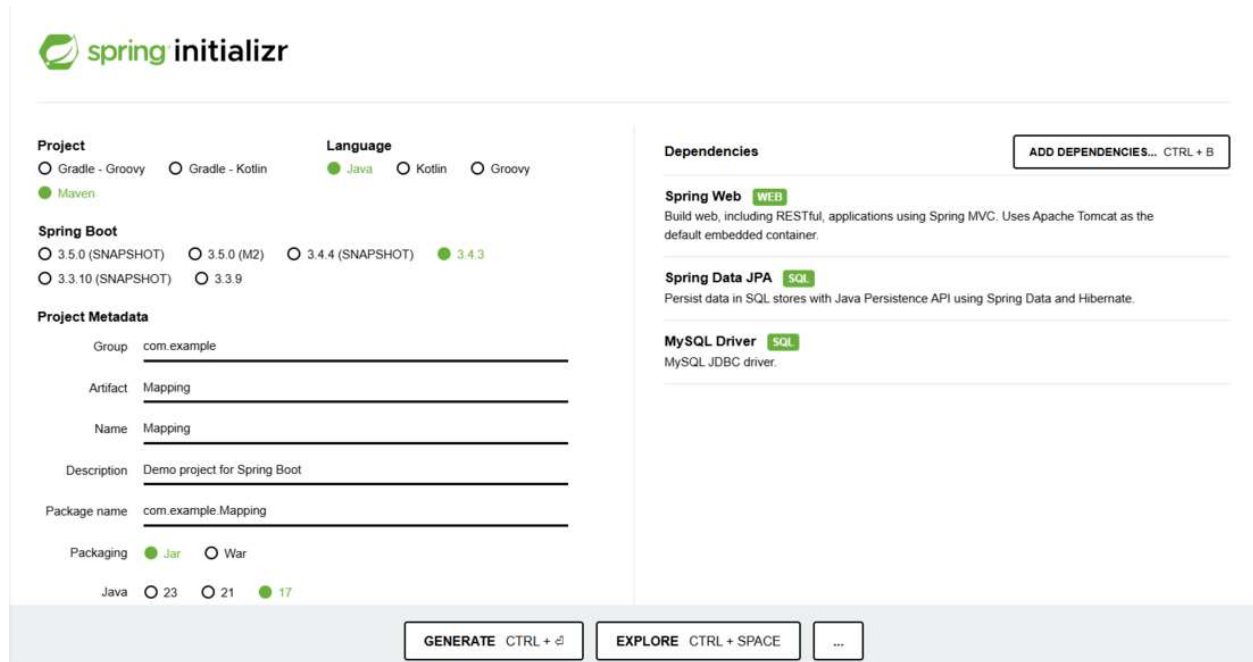
- Java 17+ (Recommended for Spring Boot 3.x)
- Spring Boot 3.x
- Spring Data JPA
- MySQL Database
- IDE (IntelliJ IDEA, Eclipse, or VS Code)

## Setting Up the Spring Boot Project

### Step 1: Create a Spring Boot Project

Go to [Spring Initializr](#) and create a new project with the following configurations:

- **Project:** Maven
- **Language:** Java
- **Spring Boot Version:** 3.x (Latest Stable version)
- **Packaging:** JAR
- **Java:** 17 or later
- **Dependencies:** Spring Web, Spring Data JPA, MySql Driver



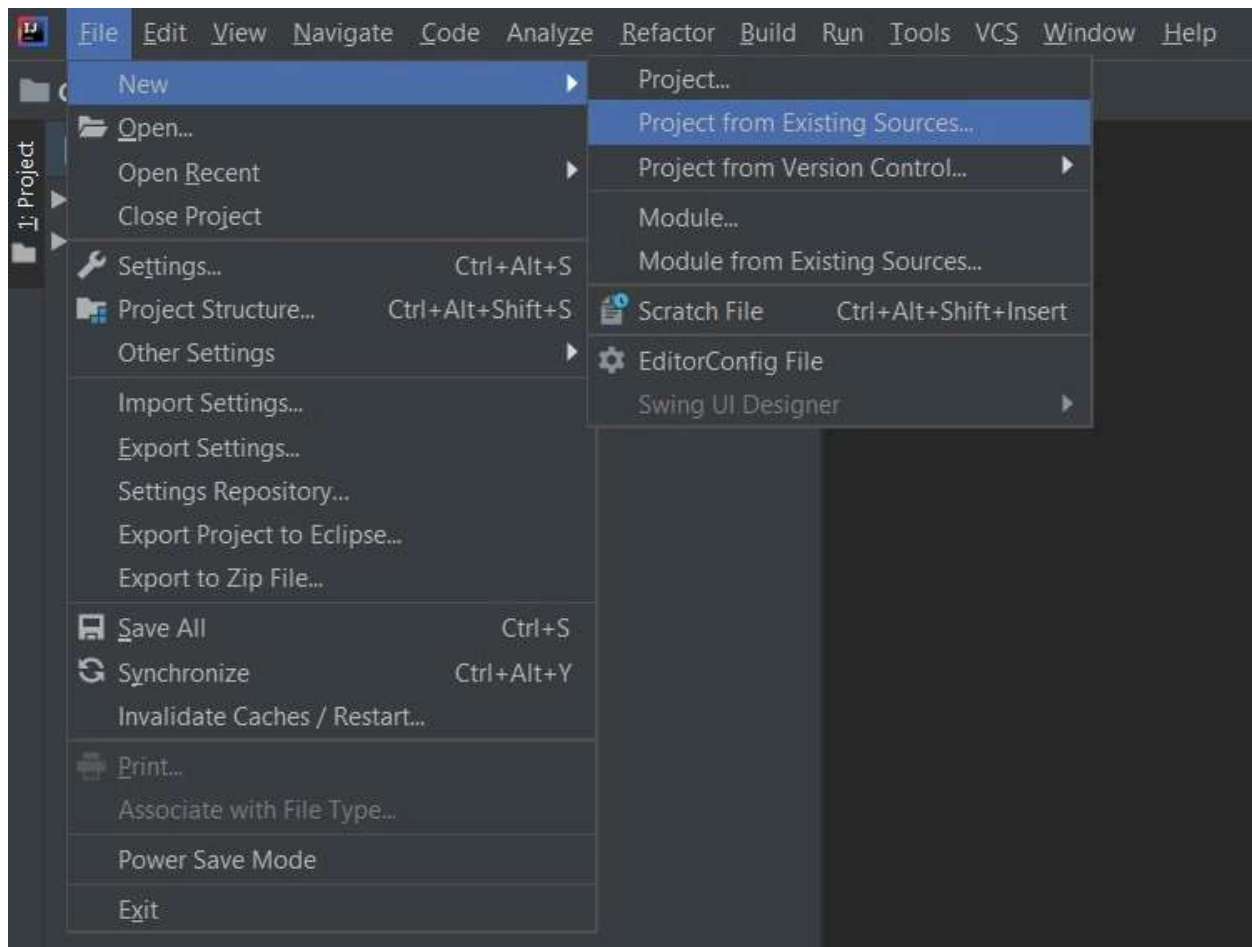
The image shows the Spring Initializr web form for configuring a new project. The form is divided into several sections:

- Project:** Includes radio buttons for **Gradle - Groovy**, **Gradle - Kotlin**, and **Maven** (which is selected).
- Language:** Includes radio buttons for **Java** (selected), **Kotlin**, and **Groovy**.
- Spring Boot:** Includes radio buttons for various versions: **3.5.0 (SNAPSHOT)**, **3.5.0 (M2)**, **3.4.4 (SNAPSHOT)**, **3.4.3** (selected), **3.3.10 (SNAPSHOT)**, and **3.3.9**.
- Project Metadata:** Includes text input fields for **Group** (com.example), **Artifact** (Mapping), **Name** (Mapping), **Description** (Demo project for Spring Boot), and **Package name** (com.example.Mapping).
- Packaging:** Includes radio buttons for **Jar** (selected) and **War**.
- Java:** Includes radio buttons for **23**, **21**, and **17** (selected).
- Dependencies:** A section on the right with a button **ADD DEPENDENCIES... CTRL + B**. It lists three dependencies: **Spring Web** (WEB), **Spring Data JPA** (SQL), and **MySQL Driver** (SQL).

At the bottom, there are three buttons: **GENERATE CTRL + G**, **EXPLORE CTRL + SPACE**, and a button with three dots.

Click on Generate which will download the starter project.

Extract the zip file. Now open a suitable IDE and then go to File > New > Project from Existing Sources and select pom.xml. Click on import changes on prompt and wait for the project to sync as pictorially depicted below as follows:

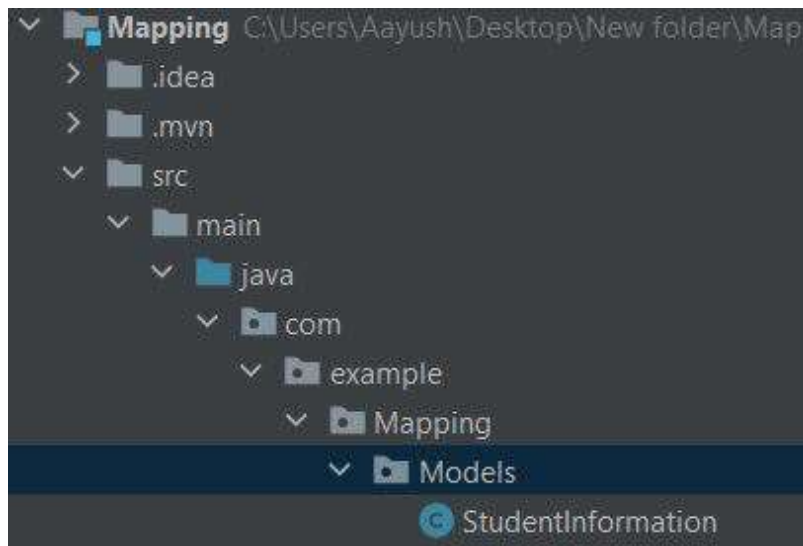


## Step 2: Configure Database in application.properties

Update the **application.properties** file with your MySQL database credentials. (mapping is the database name)

```
spring.datasource.url=jdbc:mysql://localhost:3306/mapping
spring.datasource.username=${DB_USERNAME}
spring.datasource.password=${DB_PASSWORD}
spring.jpa.hibernate.ddl-auto=update
```

## Project Structure:



## Using @Column Annotation in Entity Class

### 1. Setting Column Length

The @Column(length = n) attribute defines the maximum size of a column in the database.

**StudentInformation.java:**

```
import jakarta.persistence.*;
import lombok.*;

@Entity
@Table(name = "Student")
@Getter
@Setter
@NoArgsConstructor
@AllArgsConstructor
public class StudentInformation {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int rollno;

    @Column(name = "student_name", length = 30)
    private String name;
}
```

**Generated MySQL Table:**

```
CREATE TABLE Student (
```

```

    rollno INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(255),
    student_name VARCHAR(30)
);

```

### Run the main application:

```

: HCANN000001: Hibernate Commons Annotations {5.1.2.Final}
: HikariPool-1 - Starting...
: HikariPool-1 - Start completed.
: HHH000400: Using dialect: org.hibernate.dialect.MySQL8Dialect
: HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform]
: Initialized JPA EntityManagerFactory for persistence unit 'default'
: spring.jpa.open-in-view is enabled by default. Therefore, database queries may NOT be performed while the current application is starting.
: Tomcat started on port(s): 8080 (http) with context path ''
: Started MappingApplication in 6.117 seconds (JVM running for 6.762)

```

### Database output:

```

mysql> desc student;
+-----+-----+-----+-----+-----+-----+
| Field      | Type          | Null | Key | Default | Extra           |
+-----+-----+-----+-----+-----+-----+
| rollno     | int           | NO   | PRI | NULL    | auto_increment |
| name       | varchar(255)  | NO   |     | NULL    |                 |
| student_name | varchar(30)   | YES  |     | NULL    |                 |
+-----+-----+-----+-----+-----+-----+
3 rows in set (0.01 sec)

```

## 2. Adding the default value in the column

The recommended approach to set default values is by initializing the field.

```

@Column(nullable = false)
private String name = "Default Name";

```

### StudentInformation.java:

```

import jakarta.persistence.*;
import lombok.*;

@Entity

```



```

@Table(name = "Student")
@Getter
@Setter
@NoArgsConstructor
@AllArgsConstructor
public class StudentInformation {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int rollno;

    @Column(nullable = false)
    private String name = "Default Name";
}

```

Run the main application:

```

: HCANN000001: Hibernate Commons Annotations {5.1.2.Final}
: HikariPool-1 - Starting...
: HikariPool-1 - Start completed.
: HHH000400: Using dialect: org.hibernate.dialect.MySQL8Dialect
: HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform]
: Initialized JPA EntityManagerFactory for persistence unit 'default'
: spring.jpa.open-in-view is enabled by default. Therefore, database queries may NOT be performed while the application is starting.
: Tomcat started on port(s): 8080 (http) with context path ''
: Started MappingApplication in 6.117 seconds (JVM running for 6.762)

```

Database output:

```

mysql> desc student;
+-----+-----+-----+-----+-----+-----+
| Field | Type          | Null | Key | Default | Extra           |
+-----+-----+-----+-----+-----+-----+
| rollno | int           | NO   | PRI | NULL    | auto_increment |
| name   | varchar(255) | YES  |     | John Snow |                 |
+-----+-----+-----+-----+-----+-----+
2 rows in set (0.00 sec)

```

### 3. Adding a not-null Constraint to the Column

To enforce a NOT NULL constraint, use:

```

@Column(nullable = false)
private String name;

```



## StudentInformation.java:

```
import jakarta.persistence.*;
import lombok.*;

@Entity
@Table(name = "Student")
@Getter
@Setter
@NoArgsConstructor
@AllArgsConstructor
public class StudentInformation {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int rollno;

    @Column(nullable = false)
    private String name;
}
```

## Run the main application:

```
: HCANN000001: Hibernate Commons Annotations {5.1.2.Final}
: HikariPool-1 - Starting...
: HikariPool-1 - Start completed.
: HHH000400: Using dialect: org.hibernate.dialect.MySQL8Dialect
: HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform]
: Initialized JPA EntityManagerFactory for persistence unit 'default'
: spring.jpa.open-in-view is enabled by default. Therefore, database queries may NOT be processed while webviews are open.
: Tomcat started on port(s): 8080 (http) with context path ''
: Started MappingApplication in 6.117 seconds (JVM running for 6.762)
```

## Database Output:

```
mysql> desc student;
+-----+-----+-----+-----+-----+-----+
| Field | Type          | Null | Key | Default | Extra           |
+-----+-----+-----+-----+-----+-----+
| rollno | int           | NO   | PRI | NULL    | auto_increment |
| name   | varchar(255) | NO   |     | NULL    |                 |
+-----+-----+-----+-----+-----+-----+
2 rows in set (0.00 sec)
```

[Comment](#)
[More info](#)
[Advertise with us](#)