Spring Boot Annotation @GeneratedValue

Hrishi D · Follow

2 min read · Jan 20, 2024

(▶) Listen   (⬆) Share   (•••) More

The @GeneratedValue annotation in JPA (Java Persistence API) is used to specify the strategy used for generating primary key values for entities. The GenerationType.IDENTITY strategy is commonly used with databases that support auto-incrementing columns, such as MySQL and PostgreSQL.

Here's a breakdown of the parameters you can use with @GeneratedValue(strategy = …):

1. **GenerationType.IDENTITY:**

- This strategy relies on an auto-incremented database column for primary key generation.

- It is suitable for databases that support auto-incrementing columns (e.g., MySQL, PostgreSQL).

- The database automatically generates a unique value for the primary key when a new record is inserted.

Example:

```
ja@Id
@GeneratedValue(strategy = GenerationType.IDENTITY)
private Long id;
```

2. GenerationType.SEQUENCE:

- This strategy relies on a database sequence to generate primary key values.

- Not all databases support sequences, so it's essential to check database compatibility.

- You may need to define the sequence explicitly in your database.

Example:

```
@Id
@GeneratedValue(strategy = GenerationType.SEQUENCE, generator = "expense_sequen
@SequenceGenerator(name = "expense_sequence", sequenceName = "expense_sequence"
private Long id;
```

3. GenerationType.TABLE:

- This strategy uses a separate table to maintain a counter for generating primary key values.

- It can be less efficient than other strategies and is not commonly used.

Example:

```
@Id
@GeneratedValue(strategy = GenerationType.TABLE, generator = "expense_generator
@TableGenerator(name = "expense_generator", table = "id_generator", pkColumnNam
private Long id;
```

4. GenerationType.AUTO:

- The AUTO strategy allows the JPA provider to choose the appropriate strategy based on the underlying database capabilities.

- It's a portable way to generate primary keys, but the actual strategy used depends on the database.