Entity 기본속성

1. @Id

: 기본 테이블의 Primary Key의 Data Type

2. @GeneratedValue

```
public @interface GeneratedValue {
```

> (Optional) The primary key generation strategy that the persistence provider must use to generate the annotated entity primary key.

```
    GenerationType strategy() default AUTO;
```

```
public enum GenerationType {
```

속성값	설명	대표 DBMS
GenerationType.IDENTITY	기본키 생성을 데이터베이스에 위임	MySQL
GenerationType.SEQUENCE	시퀀스 사용, @SequenceGenerator 필요	ORACLE
GenerationType.TABLE	키 생성용 테이블 사용, @TableGenerator 필요	모든 DBMS
GenerationType.AUTO	데이터베이스 방언에 따라 자동 지정(기본값)	

- IDENTITY : Transaction이 종료되기 전에 Insert문이 동작해서 사전에 Id 값을 먼저 받아오게 된다. 실제로 Commit 되지 않고 실제로 Logic이 종료된다 하더라도 DB에 있는 ID 값을 증가시키고 있어서 이빨 빠진 것처럼 특정 Id 가 비는 현상이 생김.(MySQL, MariaDB)

- SEQUENCE : Sequence라는 특별한 함수를 제공하는 Oracle, Postgre, H2에서 사용
- TABLE : 데이터베이스와 관계없이 키 생성 테이블을 별도로 운영
- AUTO : DB 의존성 없이 자동으로 생성(Default), 일반적으로는 지정해서 사용

※ DB 선택 시 참고 사이트

- https://db-engines.com/en/

The most popular database management systems

May 2023		Score
1.	Oracle	1233
2.	MySQL	1172
3.	Microsoft SQL Server	920
4.	PostgreSQL	618
5.	MongoDB	437

» more

3. @Column

: 각 Field를 지정하는 Annotation

: Entity와 Table의 필드명을 별도 Mapping하기 위해 사용

: 기존 Legacy System에서는 주로 모음을 빼고 필드명을 부여했음.

ex> createdAt -> crtdat 등으로 표현

new User().getCrtDat 로 표현되므로 가독성이 떨어지게 됨.

```
@Column("createdat")
Private LocalDateTime createdAt;
```

```java
@Column(name = "createdat")//실제 DB Field name
private LocalDateTime createdAt;
```

```
boolean unique() default false;
```

Unique

(Optional) Whether the database column is nullable.

```
boolean nullable() default true;
```

Not Null

(Optional) Whether the column is included in SQL INSERT statements generated by the persistence provider.

```
boolean insertable() default true;
```

DML 영향 미침.

(Optional) Whether the column is included in SQL UPDATE statements generated by the persistence provider.

```
boolean updatable() default true;
```

DML 영향 미침

```
int length() default 255;
```

varChar(255) ---> default

(Optional) The precision for a decimal (exact numeric) column. (Applies only if a decimal column is used.) Value must be set by developer if used when generating the DDL for the column.

-- insertable, updatable Test

```
@Column(name = "createdat")//실제 DB Field name
private LocalDateTime createdAt;
@Column(name = "updatedat")//실제 DB Field name
private LocalDateTime updatedAt;
```

-- DDL의 경우 필드명과 Class Member변수와 스펠(대소문자) 구분이 없는 반면, DML 수행 시에는 오류가 발생함. 그래서 @Column으로 정확히 Setting 필요

3.1. insertable / updatable 테스트

```
@Entity
@Data
@AllArgsConstructor
@NoArgsConstructor
public class UserTest {
	@Id
	@GeneratedValue(strategy = GenerationType.AUTO)
	Long id;
	String name;
	String email;

	@Column(updatable = false)
	LocalDateTime insertedAt;
	@Column(insertable = false)
	LocalDateTime updatedAt;
}
```

-- 테스트 실행하면 Insert 및 Update 구문이 정상적으로 실행됨.

```
@Autowired
	UserTestRepository userTestRepository;

	@Test
	@DisplayName("user 테이블 입력")
	void inputUser() {
		UserTest userTest = new UserTest();
		userTest.setName("홍길동");
		userTest.setEmail("hong@naver.com");
		userTest.setInsertedAt(LocalDateTime.now());
		userTest.setUpdatedAt(LocalDateTime.now());
		userTestRepository.save(userTest);
	}
```

```
@Test
@DisplayName("user 테이블 수정")
void updateUser() {
    UserTest userTest1 = new UserTest();
    Optional<UserTest> userTestOptionl = userTestRepository.findById(1L);
    userTest1 = userTestOptionl.get();
    userTest1.setName("이순신");
    userTest1.setInsertedAt(LocalDateTime.now().plusDays(2));
    userTest1.setUpdatedAt(LocalDateTime.now().plusDays(2));
    userTestRepository.save(userTest1);
}
```

4. @Transient

: 실제 Table과는 연관이 없지만 객체를 위해 필요한 Member 변수인 경우 사용

: DB의 영속성과는 무관하다.

5. @Enum

ⓐ Enum Class 생성

ⓑ User Class 추가

```
public class User {
    private Gender gender;
```

ⓒ User Class 추가 - @Enumerated Annotation 반드시 삽입

```java
public class User {
    @Enumerated(EnumType.STRING)
    private Gender gender;

public @interface Enumerated {

    (Optional) The type used in mapping an enum type.
    EnumType value() default ORDINAL;
}
```

- Enumerated Type의 Default는 ORDINAL 이다.
이것은 DB에 저장될 때 Enum으로 지정된 Zero Base의 숫자가 Field에 저장되어진다.
JPA의 내부 알고리즘에 의해 꺼내질 때 Enum과 Mapping되어 실제로 보이는 부분에 있어서는 정상 작동 되어 보인다. 하지만, Service가 수정되어 Enum의 순서가 바뀌거나 추가되어 지는 경우 엉뚱한 값으로 리턴되어 질 수 있다.
그래서, 반드시 Enum 사용 시 **@Enumerated(EnumType.STRING)** Annotation을 지정해서 사용해야 한다. 필드에서도 자주 있는 실수이다.