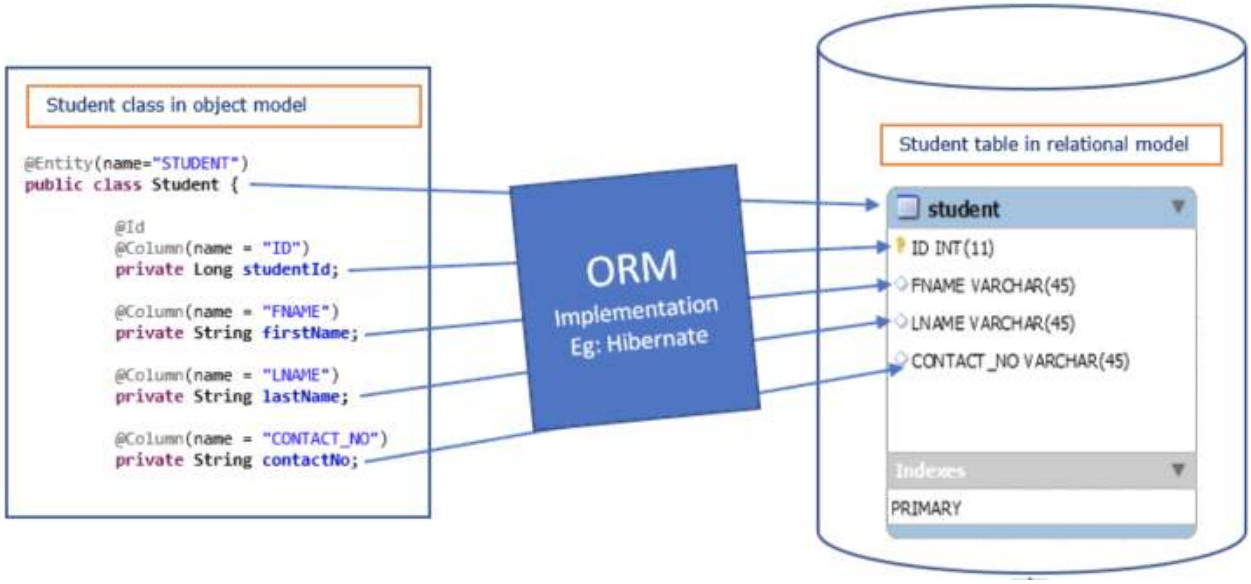
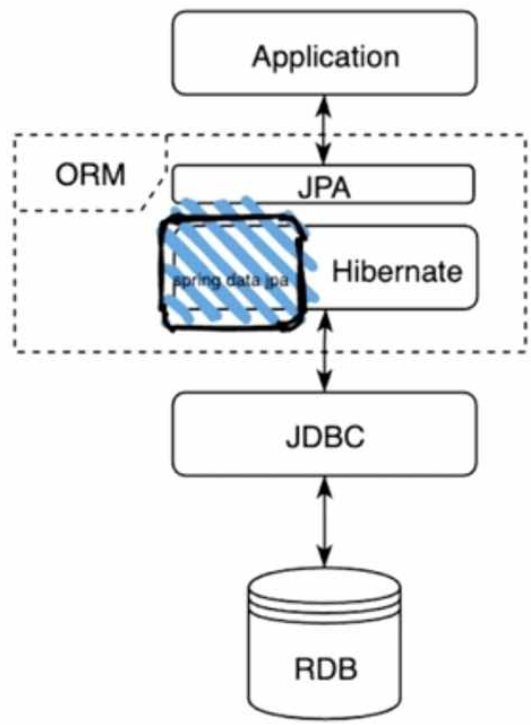
1. JPA(Java Persistence API)란?

- 현재 자바진영의 ORM(Object Relation Mapper) 표준
- Persistence 영역 데이터베이스에 접근하기 위한 API의 규격 정의
- ORM 기본 정의



- Spring JPA : 스프링에서 Hibernate를 좀더 편하게 사용할 수 있도록 추상 객체를 한번 더 감싸서 만든 것.
- Hibernate & JPA



2. 프로젝트 생성 준비

① 프로젝트 생성

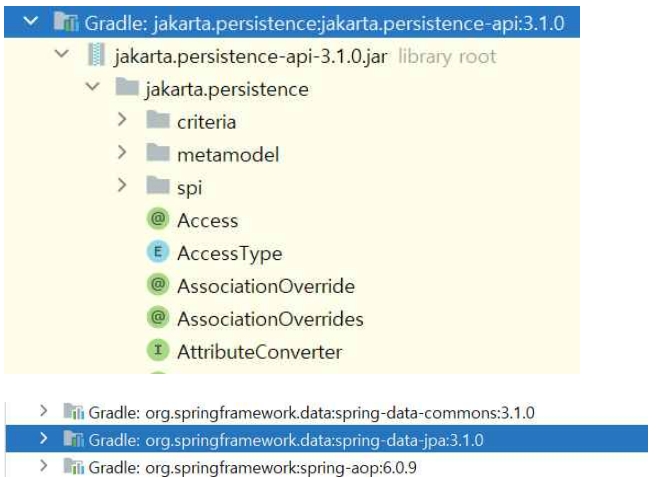
● Artifact : JpaTest

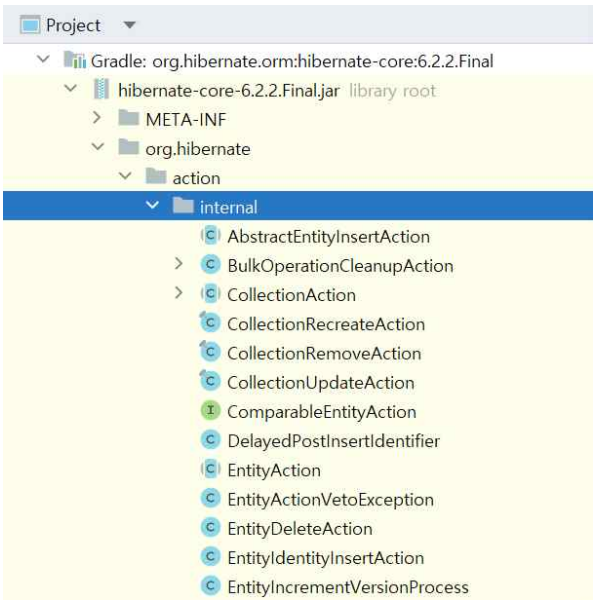
● Dependency

- Spring Boot DevTools
- Lombok
- Spring Web
- Spring Data JPA
- MySQL Driver
- H2 Database

② JPA 내부 확인해 보기

-> jakarta.persistence-api 에는 구현체는 거의 없고, interface, Enum 만 들어 있음.





-> hibernate-core 에 실제 구현체가 담겨 있음.

③ 한글 깨짐 현상 고치기(File-Settings...)

- File - Settings... - enc 검색 : UTF-8 check
- Help - Edit Custom VM Options... 에 추가

```
-Dfile.encoding=UTF-8
-DConsole.enconding=UTF-8
```

④ application.properties 설정

```
#애플리케이션 포트 설정
server.port=80

#MySQL 연결설정
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
spring.datasource.url=jdbc:mysql://localhost:3306/jpa_test
spring.datasource.username=root
spring.datasource.password=1111

#실행되는 쿼리 콘솔 출력
spring.jpa.properties.hibernate.show_sql=true

#콘솔에 출력되는 쿼리를 가독성 있게 포맷팅
spring.jpa.properties.hibernate.format_sql=true

#데이터베이스 초기화 전략- DB에 따른 SQL 구문의 차이를 해결
spring.jpa.hibernate.ddl-auto=create

#쿼리에 물음표로 출력되는 바인드 파라미터 출력
logging.level.org.hibernate.type.descriptor.sql=trace

#MySQL 방언설정
spring.jpa.database-platform=org.hibernate.dialect.MySQLDialect
```

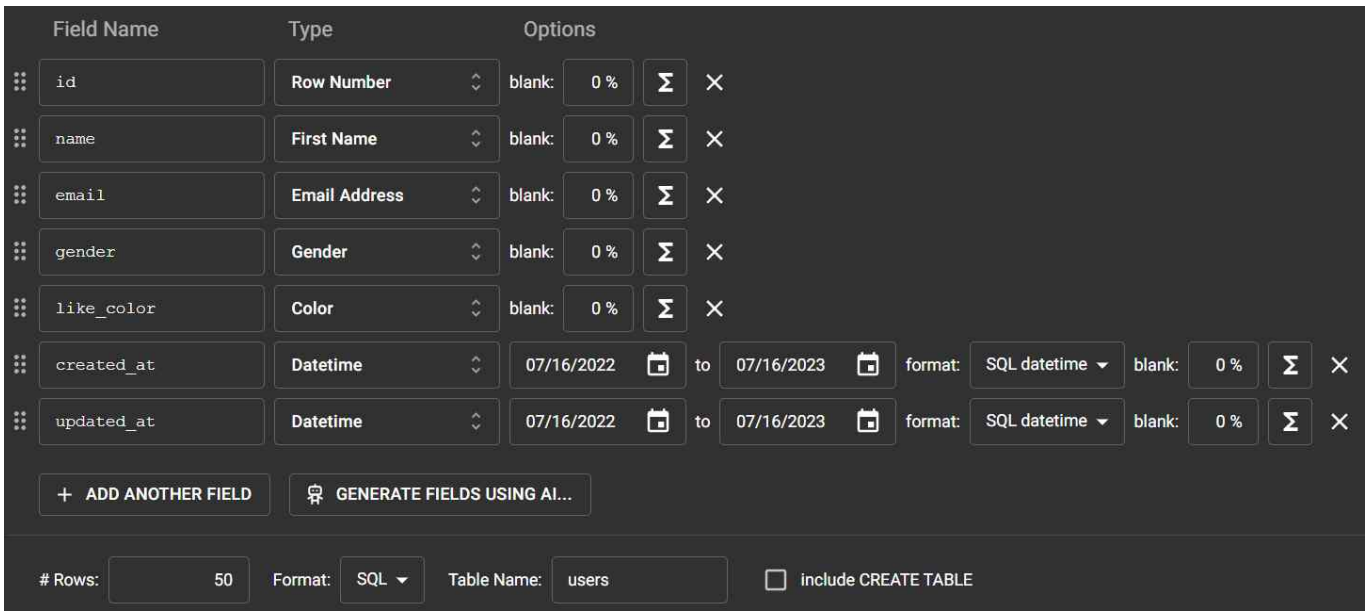
⑤ application-test.properties 생성

```
#H2 DB Setting
spring.datasource.driver-class-name=org.h2.Driver
spring.datasource.url=jdbc:h2:tcp://localhost/~/test
spring.datasource.username=sa
spring.datasource.password=

spring.jpa.database-platform=org.hibernate.dialect.H2Dialect
```

```
spring.jpa.hibernate.ddl-auto=create

#H2 DB data.sql 초기설정
spring.jpa.defer-datasource-initialization=true
sql.init.mode=always
```

⑥ dummy-data 생성

https://www.mockaroo.com/ -----> id 는 생성하지 않는다.



⑦ entity package → Users Class 생성 → UsersRepository 생성 → Test 생성

```
@Entity
@Data
@NoArgsConstructor
@AllArgsConstructor
@Builder
public class Users {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
```

```
    @Column(length = 50)
    private String name;

    @Column(length = 50)
    private String email;

    @Column(length = 50)
    private String gender;

    @Column(length = 50)
    private String likeColor;

    private LocalDateTime createdAt;
    private LocalDateTime updatedAt;
}
```

```
@SpringBootTest
@TestPropertySource(locations = "classpath:application-test.properties")
class UsersRepositoryTest {
    @Autowired
    UsersRepository usersRepository;
    @Test
    void test(){
        Long cnt = usersRepository.findAll().stream().count();
        System.out.println("----------------" + cnt);
    }
}
```

⑧ main → resources → data.sql 생성 후 다운로드 받은 insert sql 문 삽입

⑨ Test

⑩ gender member를 Enum Type 으로 변경 → constant package 생성 후 클래스 생성

```
public enum Gender {
    Male, Female
}
```

⑪ Users Class 변경

```
@Column(length = 50)
    @Enumerated(EnumType.STRING)
    private Gender gender;
```