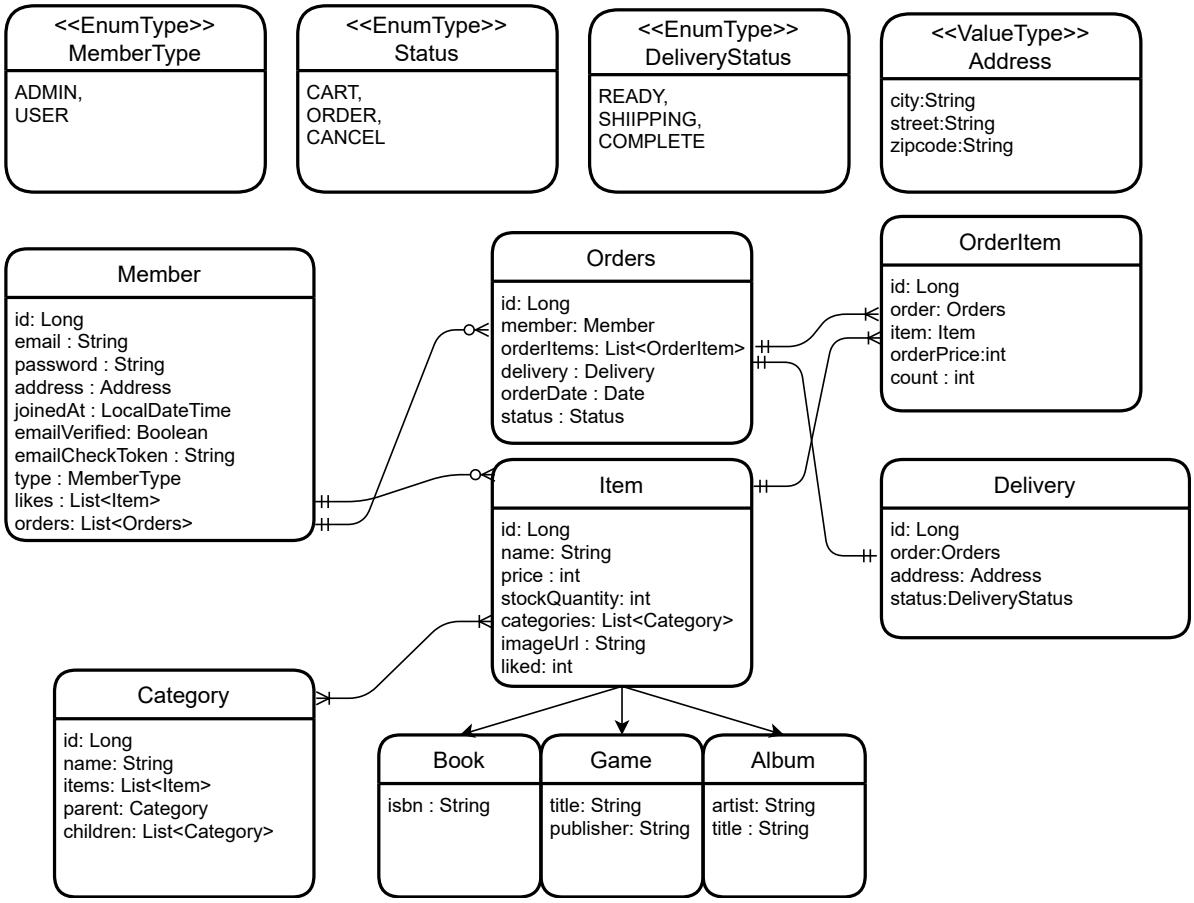
실습

요구사항

- 회원
  - 로그인
  - 회원가입
  - 이메일로 회원 인증
- 상품
  - 상품 등록
  - 상품 조회
  - 찜 순으로 조회
  - 카테고리별 조회
  - 상품명으로 조회
- 주문
  - 장바구니
  - 상품 주문
  - 상품 취소
  - 주문 내역 조회
- 기타
  - 재고 관리 (재고 추가. 재고 부족이면 주문 불가)
  - 상품 종류는 도서, 음반, 게임이 있음.
  - 카테고리는 트리 구조
  - 회원은 기본 주소지와 추가 배송 주소가 있음.

ERD



테이블 설명

MemberType

- enum
- **ADMIN, USER** 로 구성
- `Member` 엔티티에 사용됨.

Status

- enum
- **CART, ORDER, CANCEL** 로 구성
- `Orders` 엔티티에 사용됨.

DeliveryStatus

- enum
- **READY, SHIPPING, COMPLETE** 로 구성
- `Delivery` 엔티티에 사용됨.

Address

- @Embeddable 클래스
- zipcode : 우편번호
- city : 기본 주소
- street : 상세 주소
- `Member`, `Delivery` 엔티티에 사용됨.

Member

- @Entity
- email : 사용자 이메일
- password : 패스워드
- joinedAt : 가입 시간
- emailVerified : 이메일 인증 여부
- emailCheckToken : 이메일 인증 시 사용할 토큰
- type : 회원 타입 (`MemberType`)
- likes : "찜" 한 상품
- orders : 주문 내역 (`Order`)
- `Orders.member`에 사용됨.

Item

- abstract class @Entity
- 상품들(`Book`, `Album`, `Game`)의 부모클래스
- 상속 전략 : 싱글 테이블 전략
  - ```
    @Inheritance(strategy = InheritanceType.SINGLE_TABLE) <-- 요거
    @Inheritance(strategy = InheritanceType.JOINED)
    @Inheritance(strategy = InheritanceType.TABLE_PER_CLASS)
    ```
- name : 상품명
- price : 정가
- stockQuantity : 재고량
- categories : 카테고리
- imageUrl : 이미지 파일 경로
- liked : 찜 횟수
- `OrderItem.item`, `Member.likes`, `Category.items` 엔티티에 사용됨.

Album

- Item의 자식 @Entity
- @DiscriminatorValue("A")
- artist : 아티스트
- title : 앨범명

Book

- Item의 자식 @Entity
- @DiscriminatorValue("B")
- isbn : 국제표준도서번호

Game
    ```

- Item의 자식 @Entity
- @DiscriminatorValue("G")
- publisher : 배급사
- title : 게임명

Category

- @Entity
- name : 카테고리명
- items : 소속 상품
- parent : 부모 카테고리 (상위 카테고리)
- children : 자식 카테고리 (하위 카테고리)
- `Item.categoryies`에 사용됨.

Order

- @Entity
- 주문 정보
- member : 주문자
- orderItems : 주문 상품들
- delivery : 배송정보
- orderDate : 주문일자
- status : 주문 상태 (**ORDER, CANCEL**)
- `Member.orders`, `OrderItem.order`, `Delivery.order`에 사용됨.
- `Order`로 명명하지 않은 이유는 DBMS 의 ORDER 가 예약어일 수 있기 때문.

OrderItem

- @Entity
- 주문된 아이템.
- 하나의 주문에 여러 OrderItem
- Order VS Item 의 다대다 관계를 해소.
- 각 주문 상품의 **수량**과 **주문 가격**이 추가됨.
- order : 주문 정보
- item : 주문 상품
- orderPrice : 주문 시 가격
- count : 주문 수량
- `Order.orderItems`에 사용됨.

Delivery

- @Entity
- 주문 상세 정보
- order : 배송할 주문 정보
- address : 배송지
- status : 배송 상태 (**READY, SHIPPNG, COMPLETE**)

연관관계 매핑

- 회원과 찜 상품 : 일대다 단방향 (상품 입장에서 누가 찜했는지는 기록하지 않는다.)
- 회원과 주문 : 일대다 양방향
- 주문과 주문상품 : 일대다 양방향
- 주문과 배송 : 일대일 양방향
- 주문상품과 상품 : 다대일 단방향
- 카테고리와 상품 : 다대다 양방향 (일대다 양방향이지만 연습삼아..)

Address

```
package com.megait.myhome.domain;

import lombok.AllArgsConstructor;
import lombok.Getter;
import lombok.NoArgsConstructor;
import lombok.Setter;

import javax.persistence.Embeddable;

@Embeddable
@Getter
@NoArgsConstructor
@AllArgsConstructor
public class Address {

    private String zip;
    private String city;
    private String street;

}
```

Category

```
package com.megait.myhome.domain;

import lombok.Getter;
import lombok.Setter;

import javax.persistence.*;
import java.util.ArrayList;
import java.util.List;

@Entity
@Getter @Setter
public class Category {

    @Id @GeneratedValue
    private Long id;
```

```
    private String name;

    @ManyToOne(fetch = FetchType.LAZY)
    private Category parent;

    @OneToMany(mappedBy = "parent")
    private List<Category> children = new ArrayList<>();

    @ManyToMany
    @JoinTable(name="category_item",
    joinColumns = @JoinColumn(name = "category_id"),
    inverseJoinColumns = @JoinColumn(name = "item_id"))
    private List<Item> items = new ArrayList<>();

    public void addChildCategory(Category child){
        children.add(child);
        child.parent = this;
    }
}
```

Delivery

```
package com.megait.myhome.domain;

import lombok.Getter;
import lombok.Setter;

import javax.persistence.*;

@Entity
@Getter @Setter
public class Delivery {

    @Id @GeneratedValue
    @Column(name = "delivery_id")
    private Long id;

    @OneToOne(mappedBy = "delivery", fetch = FetchType.LAZY)
    private Order order;

    @Embedded
    private Address address;

    @Enumerated(EnumType.STRING)
    private DeliveryStatus status;
}
```

DeliveryStatus

```java
package com.megait.myhome.domain;

public enum DeliveryStatus {
    READY, SHIPPING, COMPLETE
}
```

Game

```java
package com.megait.myhome.domain;

import com.megait.myhome.domain.Item;
import lombok.Getter;
import lombok.Setter;

import javax.persistence.DiscriminatorValue;
import javax.persistence.Entity;

@Entity
@DiscriminatorValue("G")
@Getter @Setter
public class Game extends Item {

    private String title;
    private String publisher;

}
```

Item

```java
package com.megait.myhome.domain;

import lombok.Getter;
import lombok.Setter;

import javax.persistence.*;
import java.util.ArrayList;
import java.util.List;

@Entity
@Inheritance(strategy = InheritanceType.SINGLE_TABLE)
@Setter @Getter
public abstract class Item {
    @Id @GeneratedValue
    @Column(name = "item_id")
    private Long id;

    private String name;

    private int price;
```

```
    private int stockQuantity;

    @ManyToMany(fetch = FetchType.LAZY, mappedBy = "items")
    private List<Category> categories = new ArrayList<>();

    @Column(name = "image_url")
    private String imageUrl;

    private int liked;

}
```

Member

```
package com.megait.myhome.domain;

import lombok.EqualsAndHashCode;
import lombok.Getter;
import lombok.Setter;

import javax.persistence.*;
import java.time.LocalDateTime;
import java.util.ArrayList;
import java.util.List;

@Entity
@Getter @Setter @EqualsAndHashCode(of="id")
@Table(uniqueConstraints = {@UniqueConstraint(columnNames = {"email"})})
public class Member {

    @Id @GeneratedValue
    @Column(name = "member_id")
    private Long id;

    private String email;

    private String password;

    @Embedded
    private Address address;

    private LocalDateTime joinedAt;

    private boolean emailVerified;

    private String emailCheckToken;

    @Enumerated(EnumType.STRING)
    private MemberType type;


    @OneToMany(fetch = FetchType.LAZY)
    private List<Item> likes = new ArrayList<>();
```

```
    @OneToMany(fetch = FetchType.LAZY, mappedBy = "member")
    private List<Order> orders = new ArrayList<>();

}
```

MemberType

```
package com.megait.myhome.domain;

public enum MemberType {
    ADMIN, USER
}
```

OrderItem

```
package com.megait.myhome.domain;

import lombok.Getter;
import lombok.Setter;

import javax.persistence.*;

@Entity
@Setter @Getter
@Table(name = "order_item")
public class OrderItem {

    @Id @GeneratedValue
    @Column(name = "order_item_id")
    private Long id;

    @ManyToOne(fetch = FetchType.LAZY)
    private Order order;

    @ManyToOne(fetch = FetchType.LAZY)
    private Item item;

    @Column(name = "order_price")
    private int orderPrice;

    private int count;
}
```

Order

```java
package com.megait.myhome.domain;


import lombok.Getter;
import lombok.Setter;

import javax.persistence.*;
import java.time.LocalDateTime;
import java.util.ArrayList;
import java.util.List;

@Entity
@Getter @Setter
public class Order {

    @Id @GeneratedValue
    private Long id;

    @ManyToOne(fetch = FetchType.LAZY)
    private Member member;


    @OneToMany(mappedBy = "order", cascade = CascadeType.ALL)
    private List<OrderItem> orderItems = new ArrayList<>();

    @OneToOne(fetch = FetchType.LAZY, cascade = CascadeType.ALL)
    private Delivery delivery;

    private LocalDateTime orderDate;

    @Enumerated(EnumType.STRING)
    private Status status;

    public void setMember(Member member){
        this.member = member;
        member.getOrders().add(this);
    }

    public void addOrderItem(OrderItem orderItem){
        orderItems.add(orderItem);
        orderItem.setOrder(this);
    }

    public void setDelivery(Delivery delivery){
        this.delivery = delivery;
        delivery.setOrder(this);
    }
}
```

Status

```java
package com.megait.myhome.domain;

public enum Status {
    CART, ORDER, CANCEL
}
```