2022-02 Database Design
Report IV

중앙대학교 소프트웨어학부
20193802 조명근

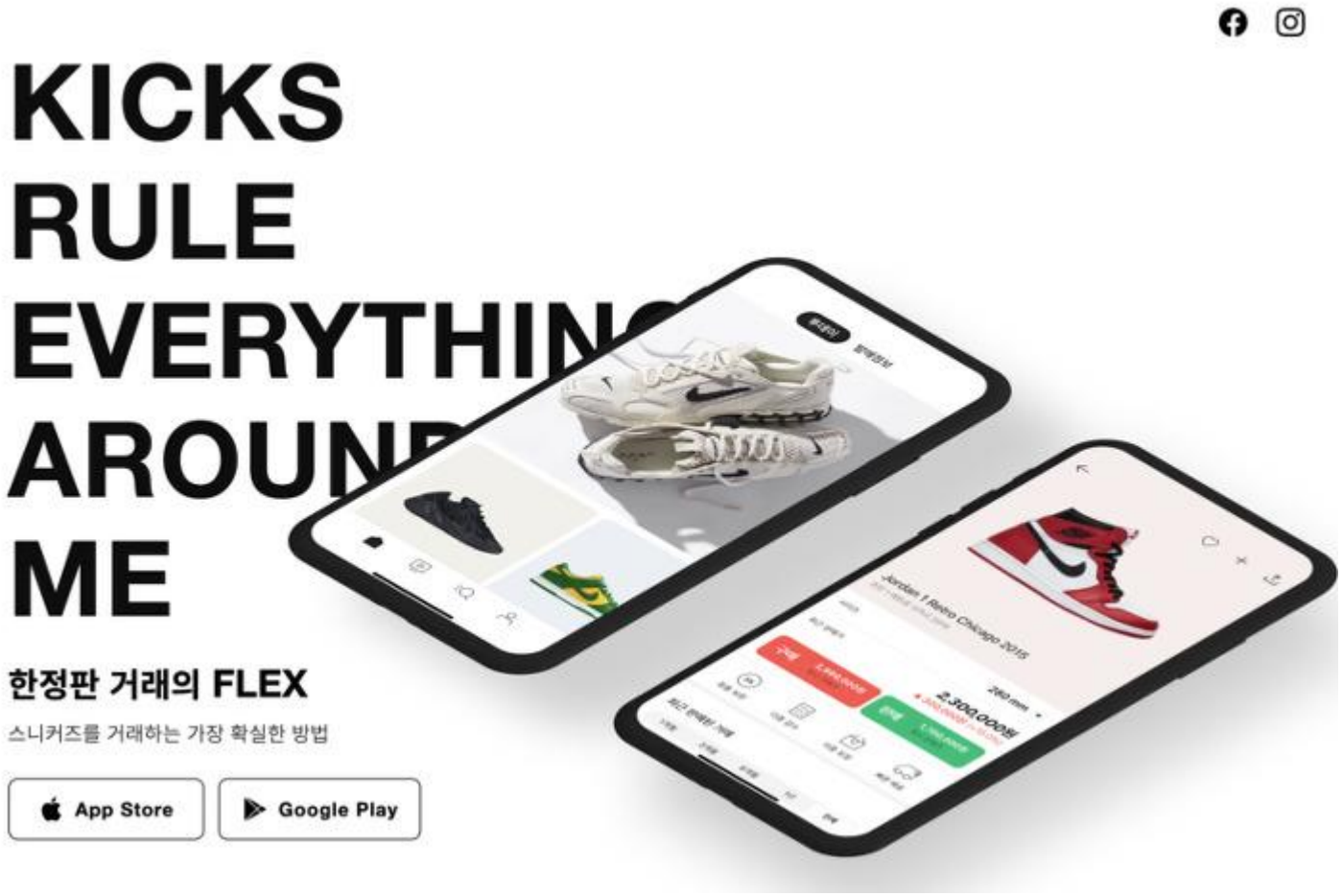
목차

(1) 응용분야 제목

응용분야에 대한 소개

(1) 응용분야 제목

온라인 한정판 거래 플랫폼



-> 한정판 발매정보 제공

-> 제품을 착용한 사진을 인스타그램처럼 갤러리화(SNS기능 제공)

-> 실시간으로 입찰을 받아서 제품의 구매입찰 가격과 판매입찰 가격이 update되고 이를 토대로 거래내역 및 현재 가격을 볼 수 있는 구조

(2) ERD

ERD 및 작성에 대한 설명

ERD

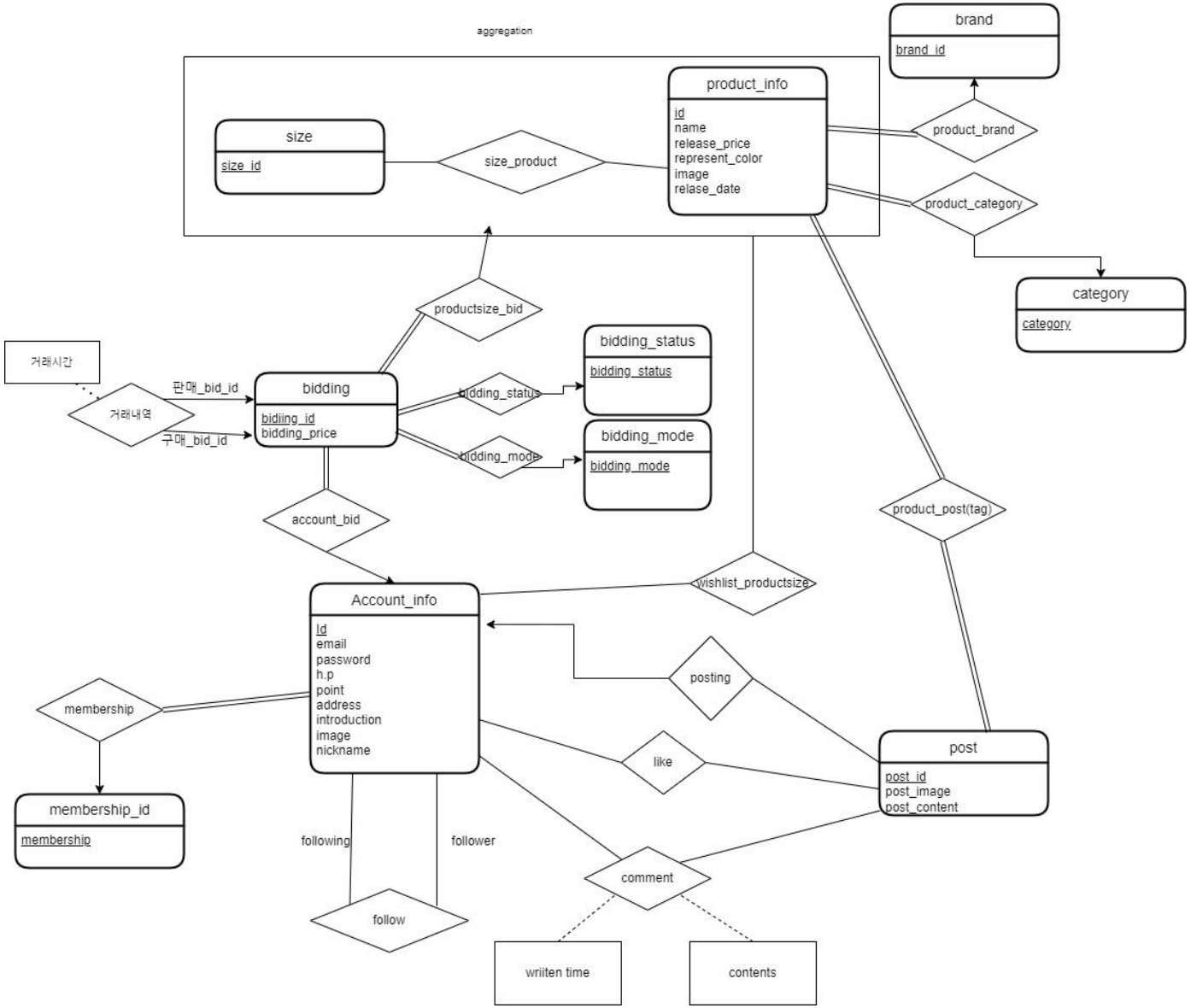
온라인 한정판 거래 플랫폼은
1. 한정판 발매정보 제공

2. 제품을 착용한 사진을 인스타그램처럼 갤러리화(SNS기능 제공)

3. 쇼핑몰과 같이 제품을 구매할 수 있으나, 차별화 된 점은 구매자이면서 판매자가 될 수 있습니다.
->실시간으로 입찰을 받아서 제품의 구매입찰 가격과 판매입찰 가격을 update해주고, 사용자에게 중계를 해줍니다.

따라서
크게 **사용자 정보, 제품 정보, 입찰 내역에 대한 정보, 거래 내역 정보, sns글에 대한 정보**를 DB에 저장해야합니다.

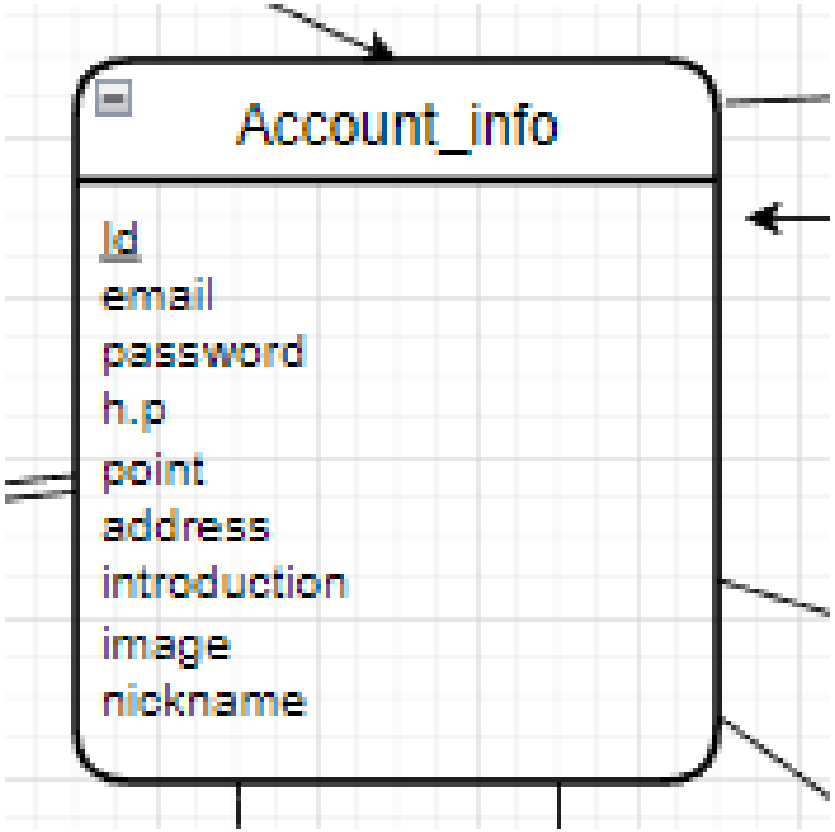
ERD


aggregation
brand
brand_id
product_info
id
name
release_price
represent_color
image
relase_date
size
size_id
size_product
product_brand
product_category
category
category
productsize_bid
bidding_status
bidding_status
bidding_status
bidding_mode
bidding_mode
거래시간
거래내역
판매_bid_id
구매_bid_id
bidding
bidiing_id
bidding_price
account_bid
product_post(tag)
wishlist_productsize
Account_info
Id
email
password
h.p
point
address
introduction
image
nickname
posting
membership
membership_id
membership
like
post
post_id
post_image
post_content
following
follower
follow
comment
wriiten time
contents

ERD

1. 사용자 정보

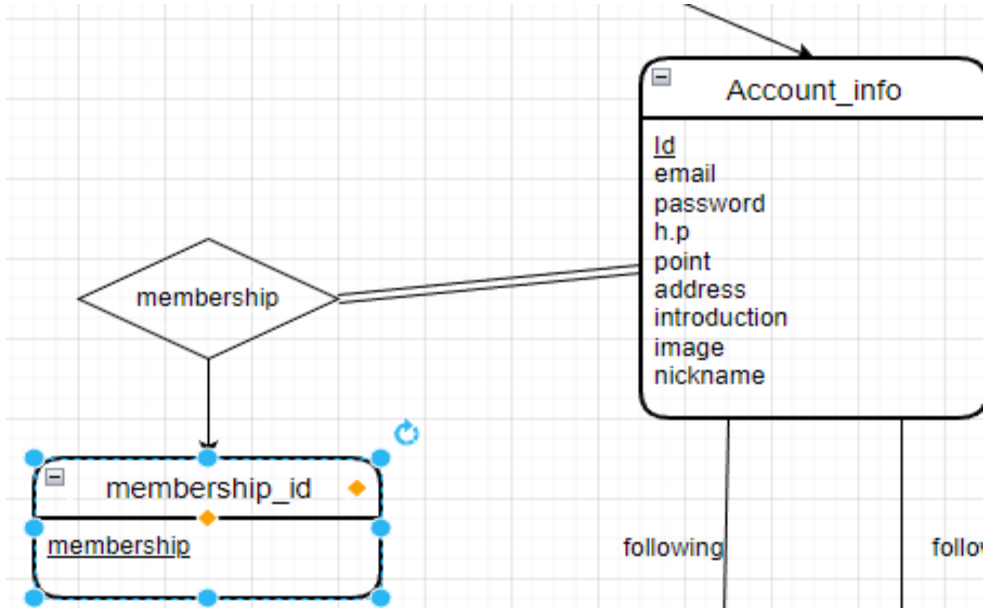
사용자 정보를 바탕으로 모든 거래 및 SNS기능을 수행합니다.



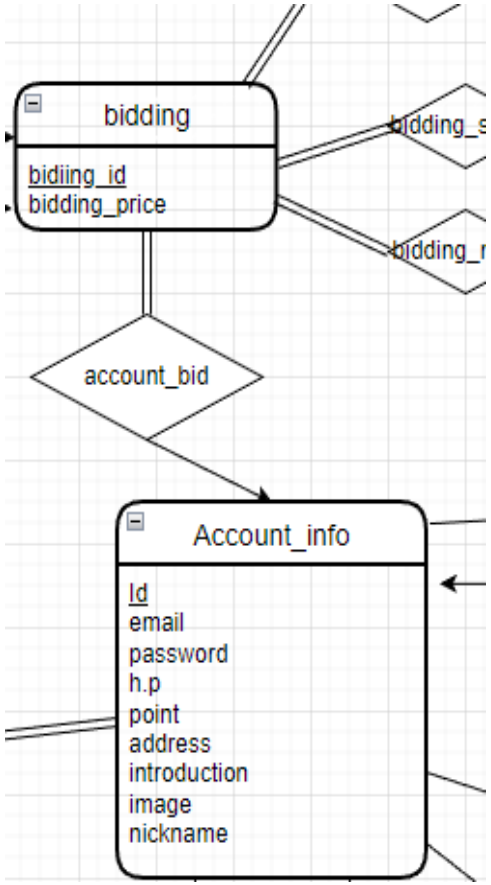
사용자 정보 entity set는 id,email ,release_price,represent_color, image ,release_date , nickname으로 이루어져있습니다.
또한 사용자 정보의 relation은
SNS, 입찰거래, 관심상품 의 관계를 구성합니다.

ERD

1. 사용자 정보
입찰 거래 및 account_info와 membership entity relation sets



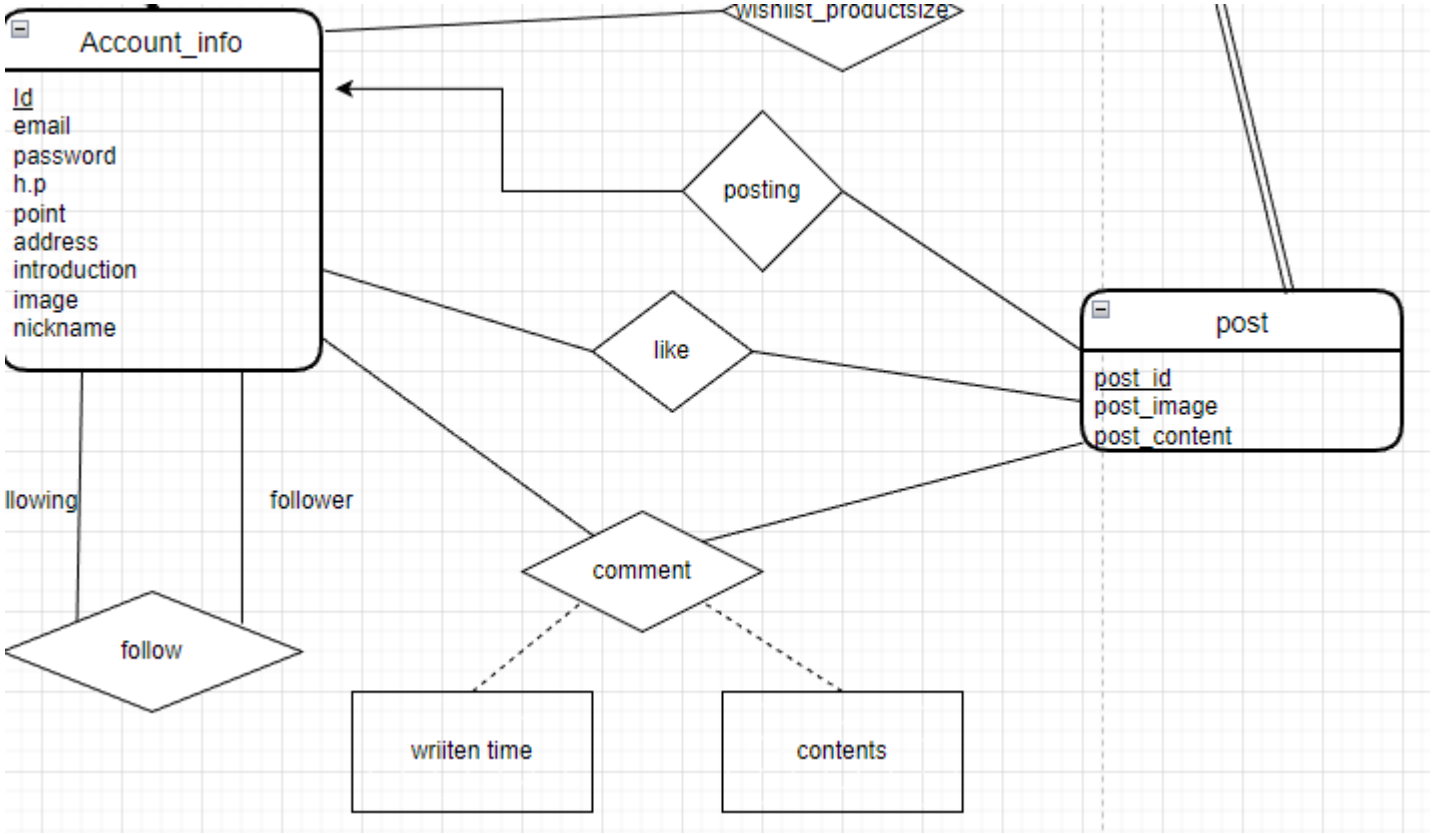
membership entity와 사용자 정보는 1:M관계에 있습니다



사용자와 bidding(입찰)은 1:M의 관계가 있습니다. 개인이 여러 개의 입찰을 진행할 수 있기 때문입니다.

ERD

1. 사용자 정보
SNS 기능 수행 relation.



SNS기능 구현을 위해 여러 relation set이 존재합니다.
1. Follow relation은 following과 follower를 unary로 연결하였고, M:N관계입니다

2. Comment relation은 사용자와 post를 M:N 관계로 연결해줍니다. (여러 개의 post에 여러명의 사용자가 여러 개의 댓글)

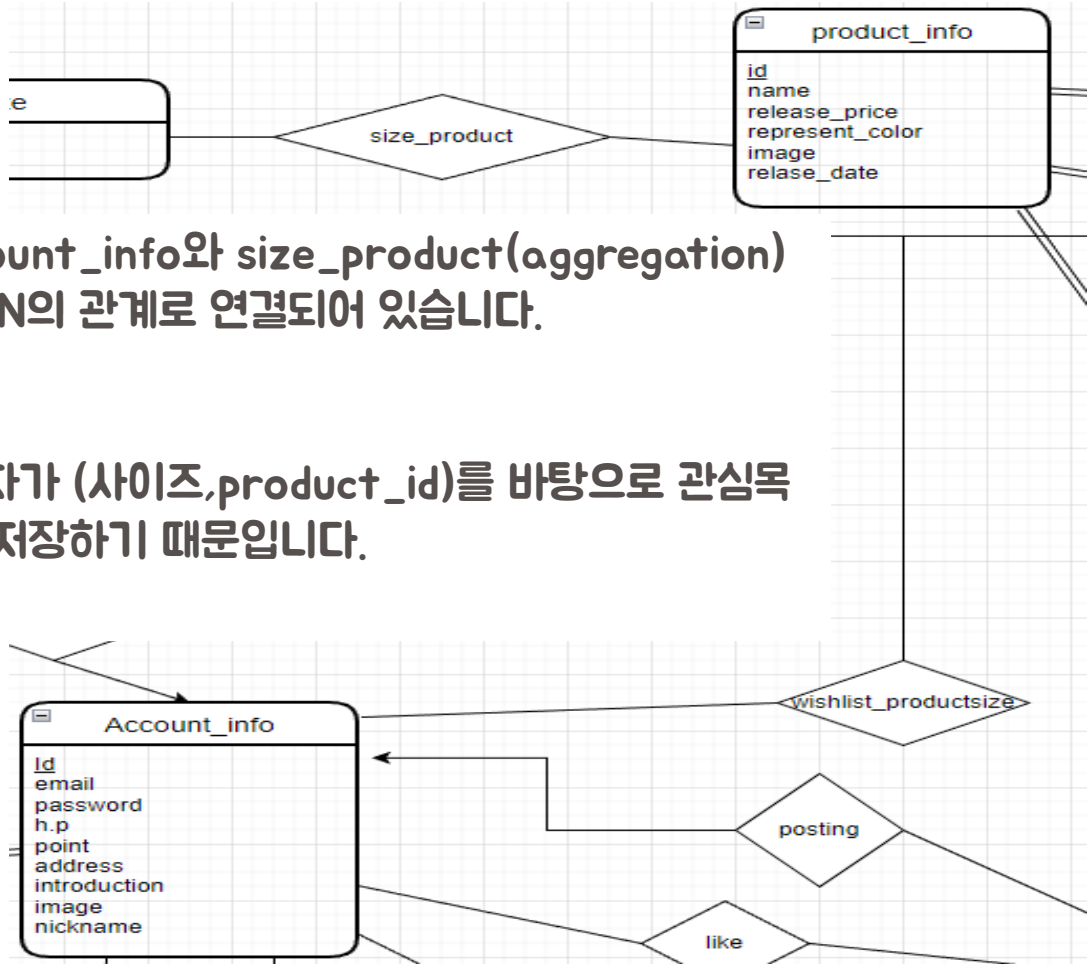
3. Posting relation은 사용자가 post를 작성하는 관계입니다. 1:M관계로 연결해줍니다.

4. Like relation은 사용자와 post를 M:N관계로 연결하는 관계입니다 (여러 명의 사용자가 여러 개의 post에 좋아요 표시)

ERD

1. 사용자 정보

제품정보와 size를 담은 aggregation set와 사용자 정보를 바탕으로 wishlist 관계 완성



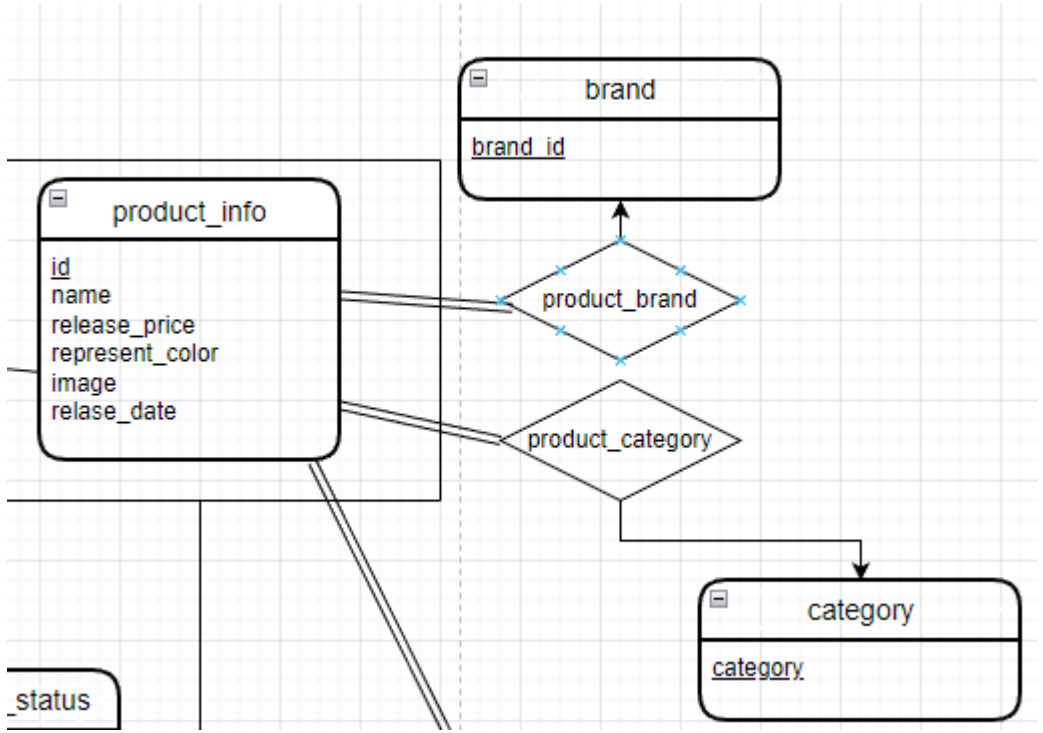
Account_info와 size_product(aggregation)는 M:N의 관계로 연결되어 있습니다.

사용자가 (사이즈,product_id)를 바탕으로 관심목록을 저장하기 때문입니다.

ERD

2. 제품 정보

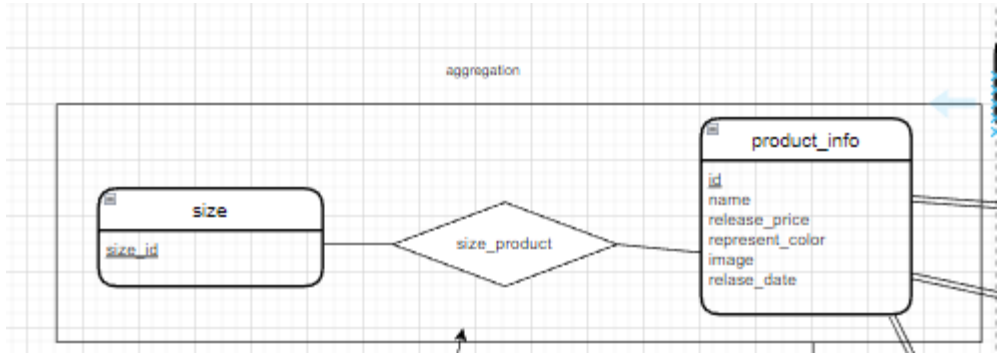
제품에 관한 정보를 erd에 저장합니다.



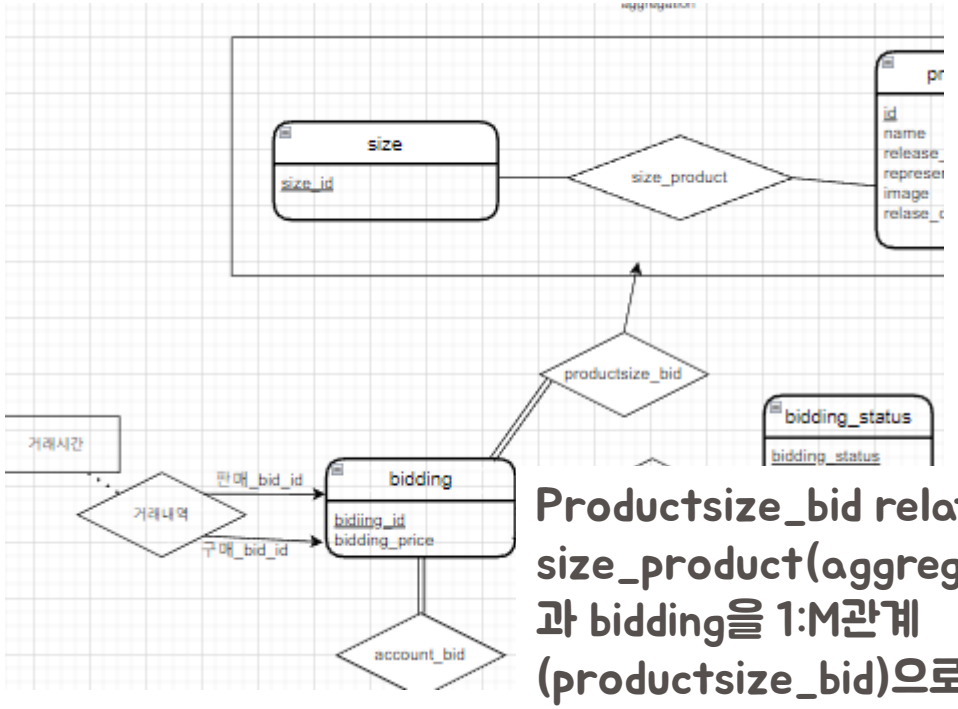
제품 정보 entity set는
id,name,release_price,represent_color,
image ,release_date으로 이루어져있습니다.
Category와 brand entit와 product_info는
1:M 관계를 맺습니다.

ERD

2. 제품 정보
Size와 product에 관한 relation



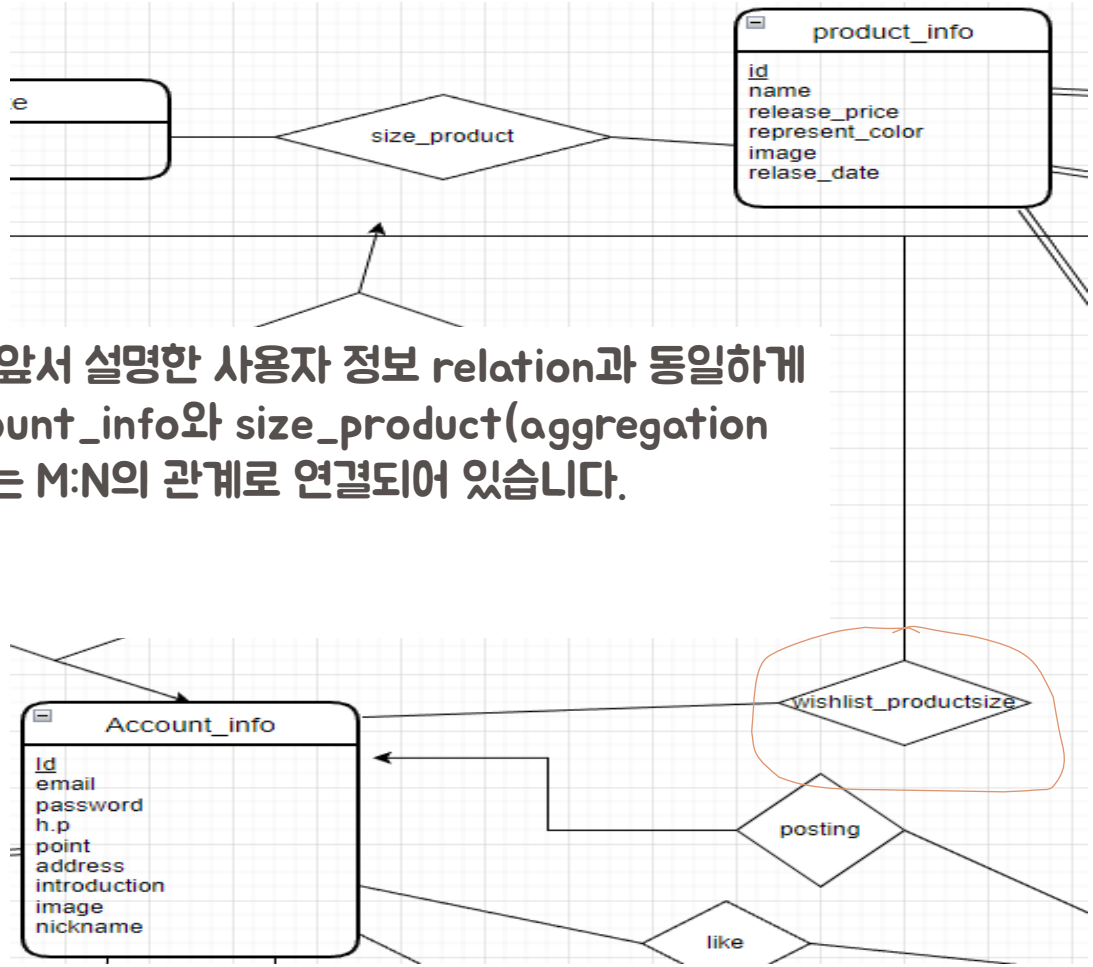
Size_product relation 은 size와 product_info의 관계를 M:N으로 표현합니다. 입찰에서 제품과 size를 바탕으로 거래하기 때문에 size와 product_info를 연결한 aggregation set로 사용해야 합니다.



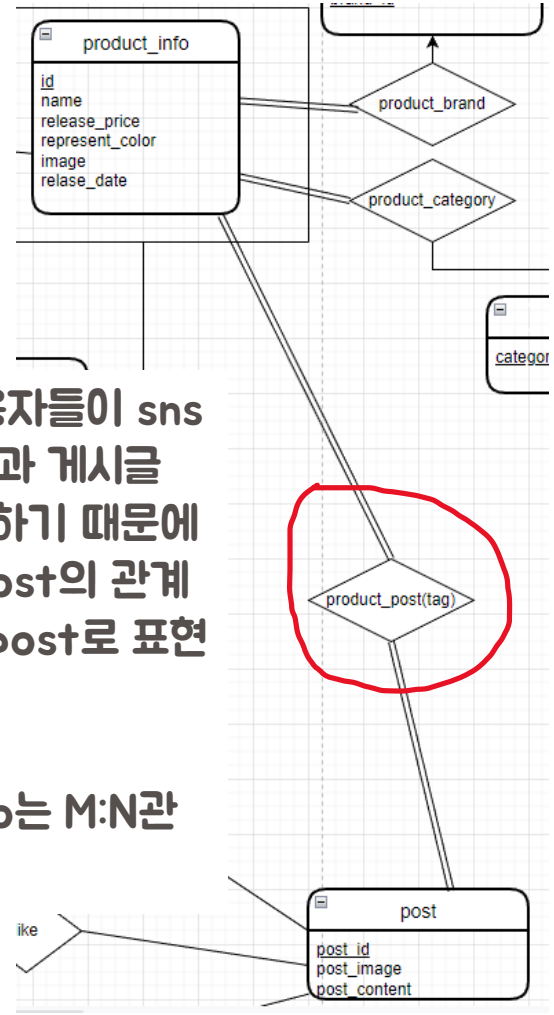
Productsize_bid relation은 size_product(aggregation set)과 bidding을 1:M관계(productsize_bid)으로 연결합니다

ERD

2. 제품 정보
Size와 product에 관한 relation



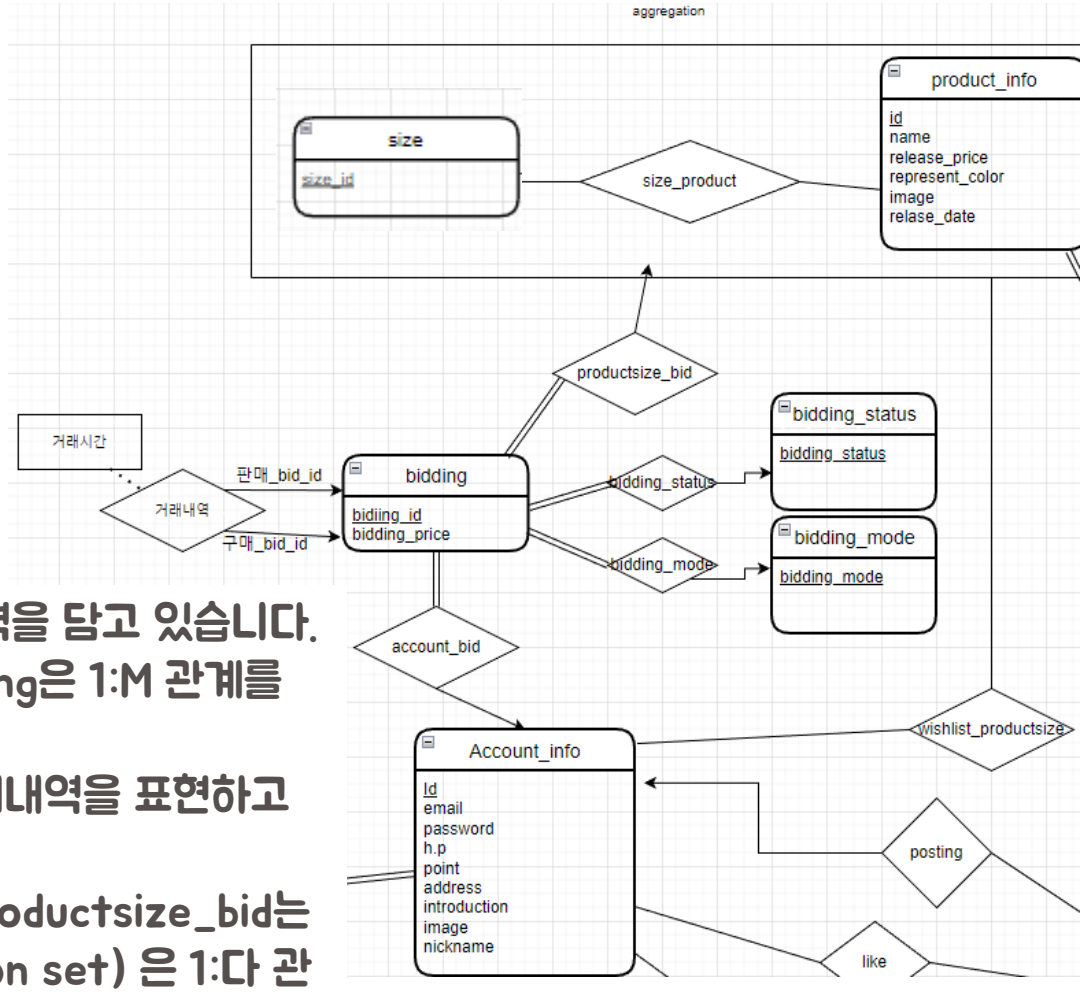
또한 앞서 설명한 사용자 정보 relation과 동일하게 Account_info와 size_product(aggregation set)는 M:N의 관계로 연결되어 있습니다.



또한 많은 사용자들이 sns 기능에서 상품과 게시글 이미지를 태그하기 때문에 product와 post의 관계를 product_post로 표현했습니다.
이때 post와 product_info는 M:N관계입니다

ERD

3. 입찰정보
Size와 product에 관한 relation

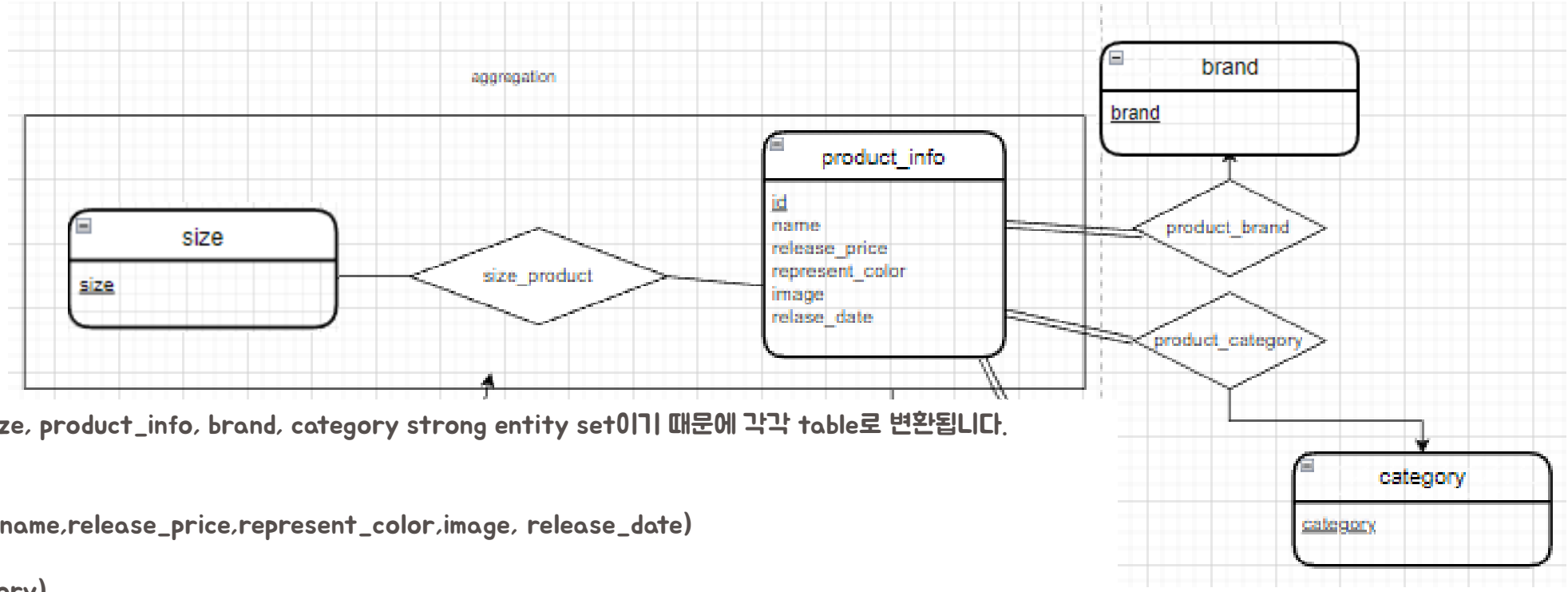


Bidding entity set는 입찰거래 id 와 입찰가격을 담고 있습니다.
또한 bidding_status, bidding_mode와 bidding은 1:M 관계를 이루고 있습니다.
또한 거래내역을 unary relatio으로 하여 거래내역을 표현하고 있습니다.
앞서 설명했던 account_info에서 설명했던 productsize_bid는 account_info와 size_product(aggregation set) 은 1:다 관계를 이루고 있습니다.

(3) RDB

ERD to RDB 변환 규칙의 적용을 통한 스키마 도출

ERD to RDB



Product_info, size, product_info, brand, category strong entity set이기 때문에 각각 table로 변환됩니다.

size(size)
Product_info(id,name,release_price,represent_color,image, release_date)
Brand(brand)
Category(category)

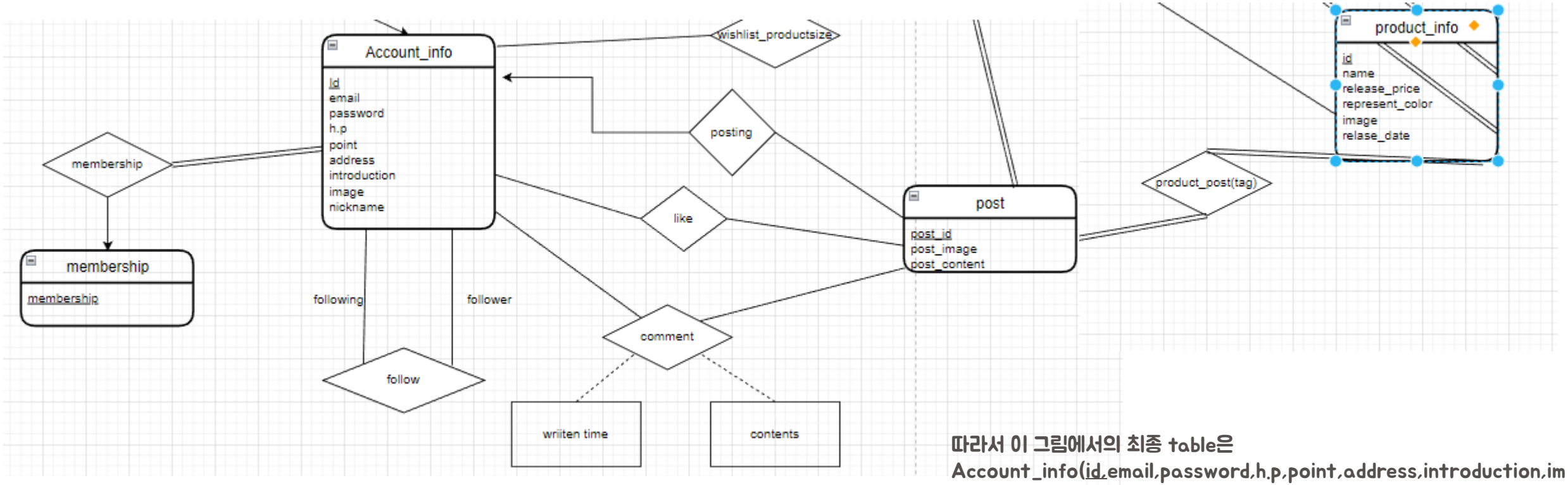
이 때, brand, category와 product_info가 1:M relation이기 때문에

Product_info(id,name,release_price,represent_color,image, release_date,brand,category)이 됩니다.

또한 size_product relationship set는 M:N relation이기 때문에
Size_product(size,id) table또한 추가됩니다.

따라서 이 그림에서의 최종 table은
size(size)
Product_info(id,name,release_price,represent_color,image, release_date,brand,category)
Brand(brand)
Category(category)
Size_product(size,id)

ERD to RDB

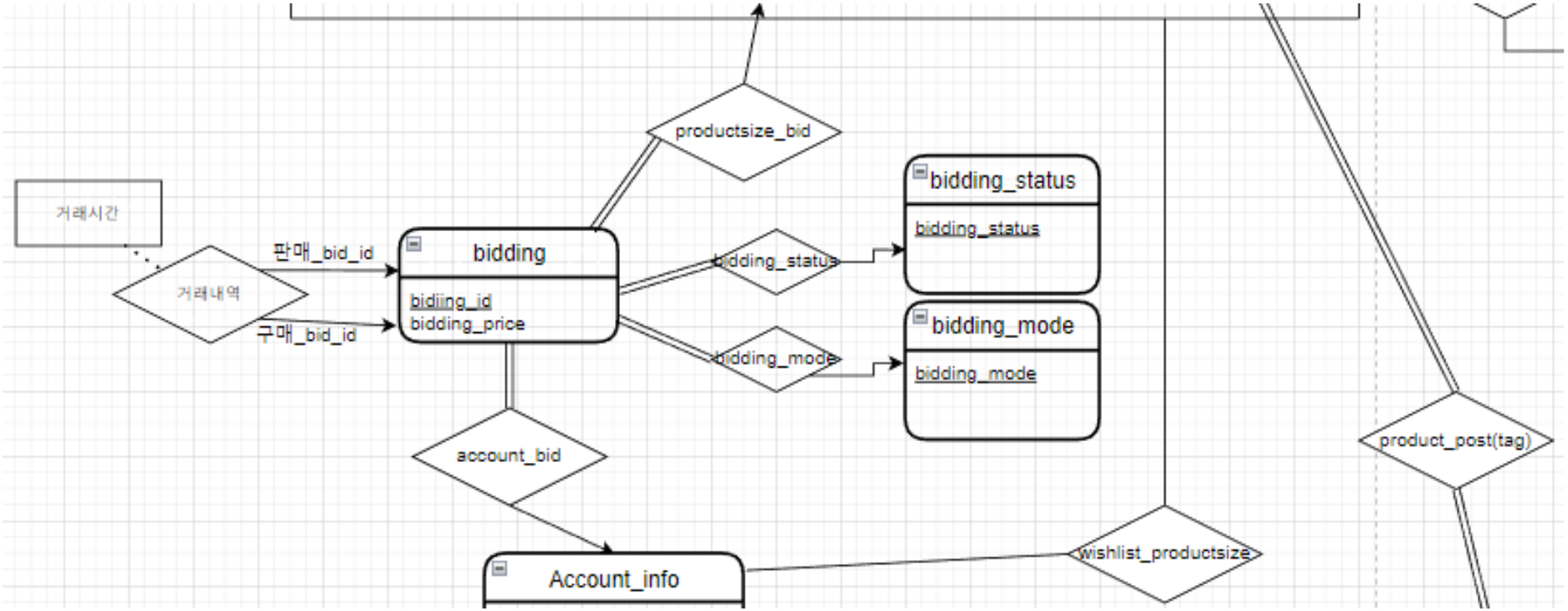


Account_info , membership, post모두 strong entity set이기 때문에
Account_info(id,email,password,h.p,point,address,introduction,image,nickname)
Membership(membership)
Post(post_id,post_image,post_content)

이때 membership과 account_info가 1:M 관계이므로 account_info table에 membership을 추가해줍니다.
- follow relation set는 account_info와 unary-M:N관계이므로 follow(following,follower)이 됩니다.
- Comment relation set은 account_info와 post가 M:N관계이므로 comment(id,post_id,written_time, contents)
- Like relation set은 account_info와 post가 M:N관계이므로 like(id,post_id)
- posting relation set은 account_info와 post가 1:M관계이므로 post table에 id를 추가해줍니다.
- Product_post relation set은 post와 product_info를 M:N관계이므로 product_post(post_id, product_id)

따라서 이 그림에서의 최종 table은
Account_info(id,email,password,h.p,point,address,introduction,image,nickname,membership)
Membership(membership)
Post(post_id,post_image,post_content,id)
follow(following,follower)
comment(id,post_id,written_time, contents)
like(id,post_id)
Product_post(post_id, product_id)

ERD to RDB

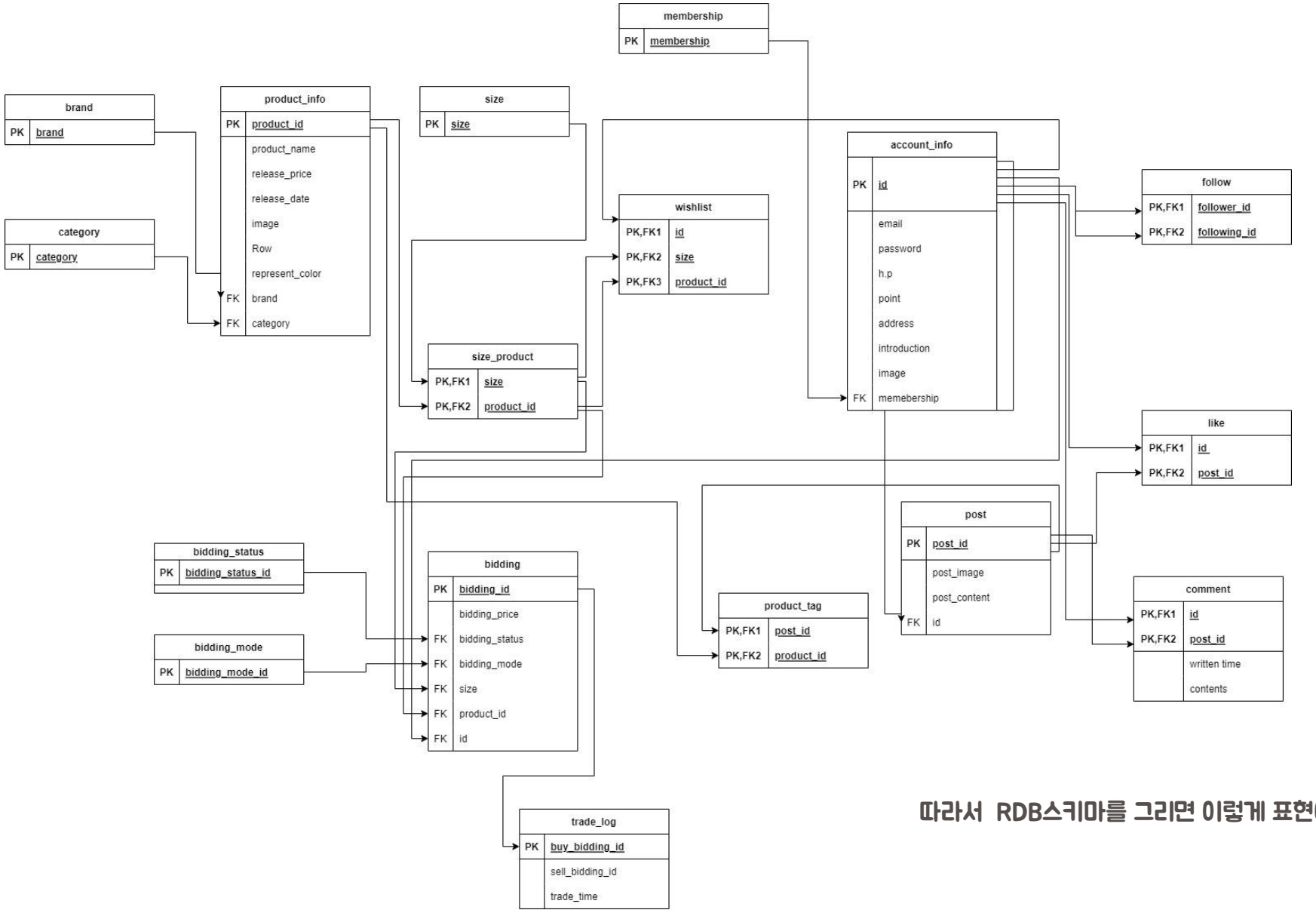


Bidding, bidding_status, bidding_mode는 strong entity set이기 때문에
Bidding(bidding_id,bidding_price)
bidding_status(bidding_status)
Bidding_mode(bidding_mode)

이때 거래내역 relation set → unary 1:1 relationship 따라서 (buy_bid_id, sell_bid_id, trade_time)
또한 account_bid, biddig_status, bidding_mode, productsize_bid relation set 과 연결된 table들이 모두 bidding entity와 1:다 관계이므로 bidding(bidding_id, bidding_price, **bidding_status, bidding_mode, id, size, product_id**)
또한 wish_productsize relationset은 account_info와 size_product table을 M:N으로 연결하기 때문에
Wishlist(id, size, product_id)

따라서 이 그림에서의 최종 table은
bidding(bidding_id, bidding_price, **bidding_status, bidding_mode, id, size, product_id**)
bidding_status(bidding_status)
Bidding_mode(bidding_mode)

ERD to RDB



따라서 RDB스키마를 그리면 이렇게 표현이 가능합니다.

(4) SQL example

SQL example

1. 가장 많이 거래된 제품 순으로 정렬하여 카테고리 별로 제품이름 , 제품 브랜드, 이미지, 제품 거래 횟수를 가져오는 SQL

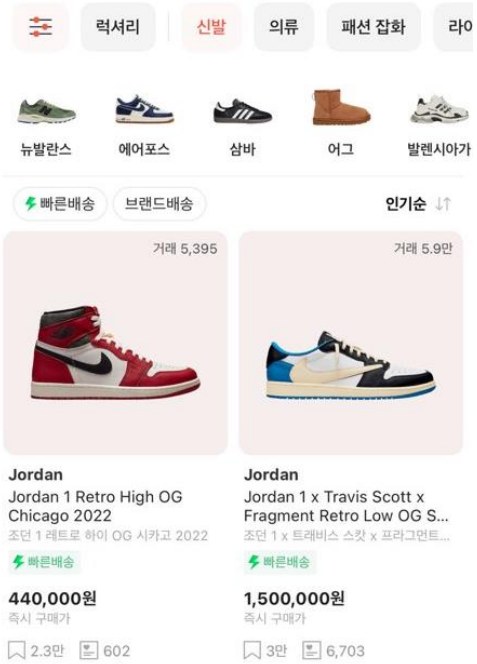
```
Select product_name, brand , count(*) as trade_count, image
From product_info as A , bidding as B
Where A.product-id = B.product_id and A.category = {user_input}
Group by product_id
Order by count(*) DESC
```

2. 계정의 아이디를 검색하면 아이디의 팔로워 명수를 가져오는 SQL

```
Select count(*) as follower_amount
From follow
Where follower_id = {user_input}
Group by (follower_id)
```

3. Nickname으로 wishlist의 값을 가져오는 SQL

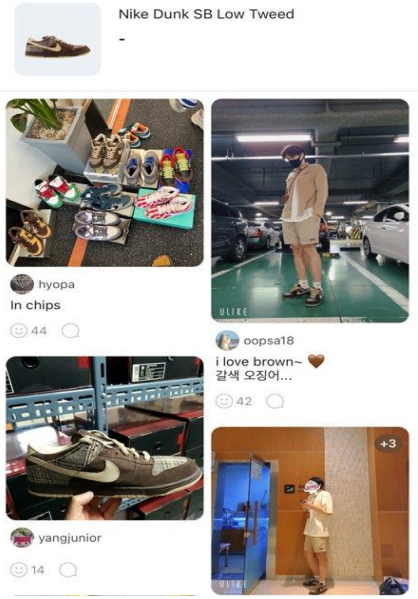
```
Select size, product_id
From wishlist
Where id in (select id from account_info where nickname = {user_input})
```



SQL example

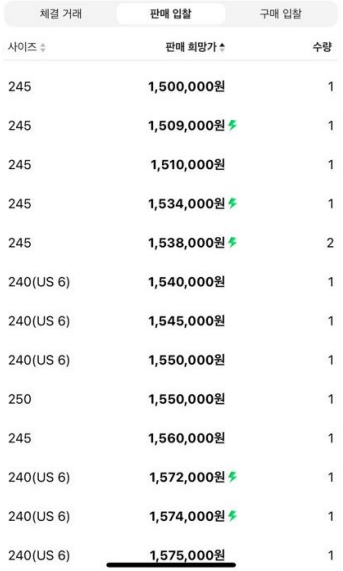
4. 제품이름을 바탕으로 제품을 태그한 게시글을 검색한 결과 값을 제공해주는 SQL

```
Select post_id , post_image
From post
Where post_id in ( select post_id
                   from product_tag
                   where product_id in ( select product_id
                                         from product_info
                                         where product_name = {user_input}))
```



5. 원하는 구매, 판매 입찰 모드에서 사이즈 별 입찰가격의 현재 현황과 그 개수의 결과 값을 제공해주는 SQL

```
Select size, bidding_price , count(*) as count_of_trade
From bidding
Where bidding_mode = '{user_input}' and product_id in (select product_id
                                                       from product_info
                                                       where product_name = {user_input})
Groupby (bidding_price , size)
Orderby (count_of_trade)
```

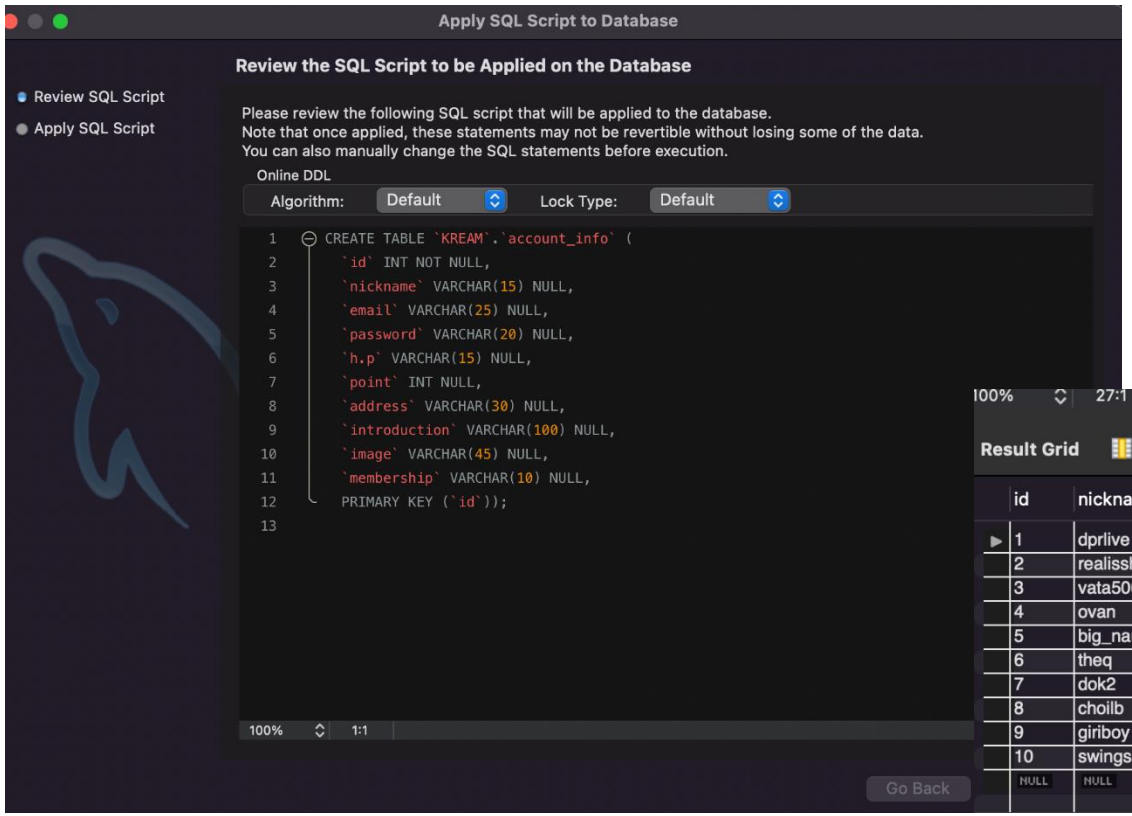


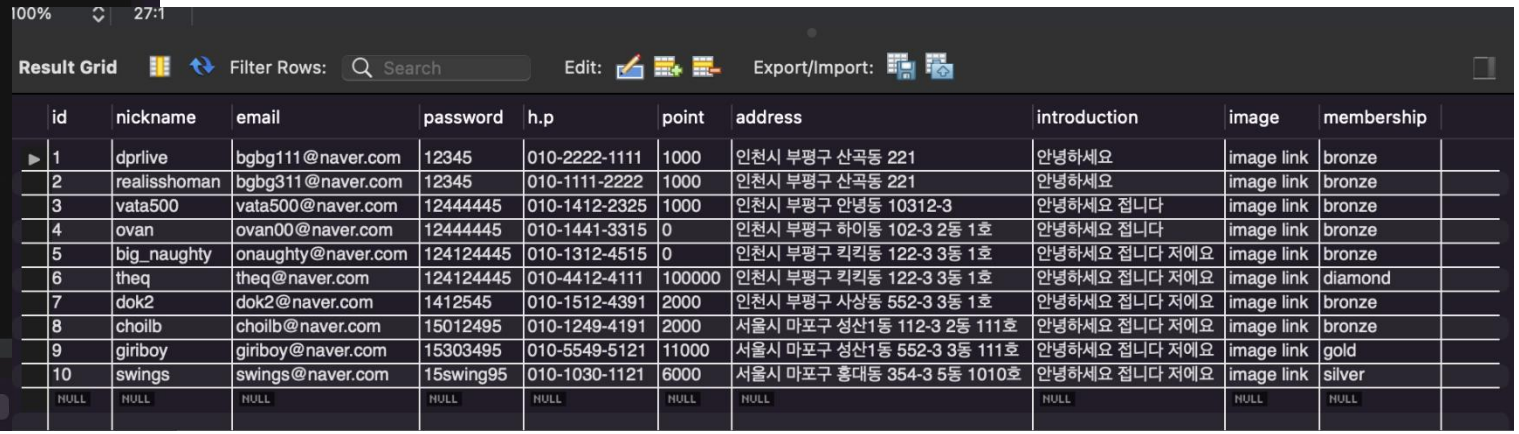
(5) DB생성 및 데이터 적재

By workbench

DB생성 및 데이터 적재

1. 계정 정보를 담은 account_info

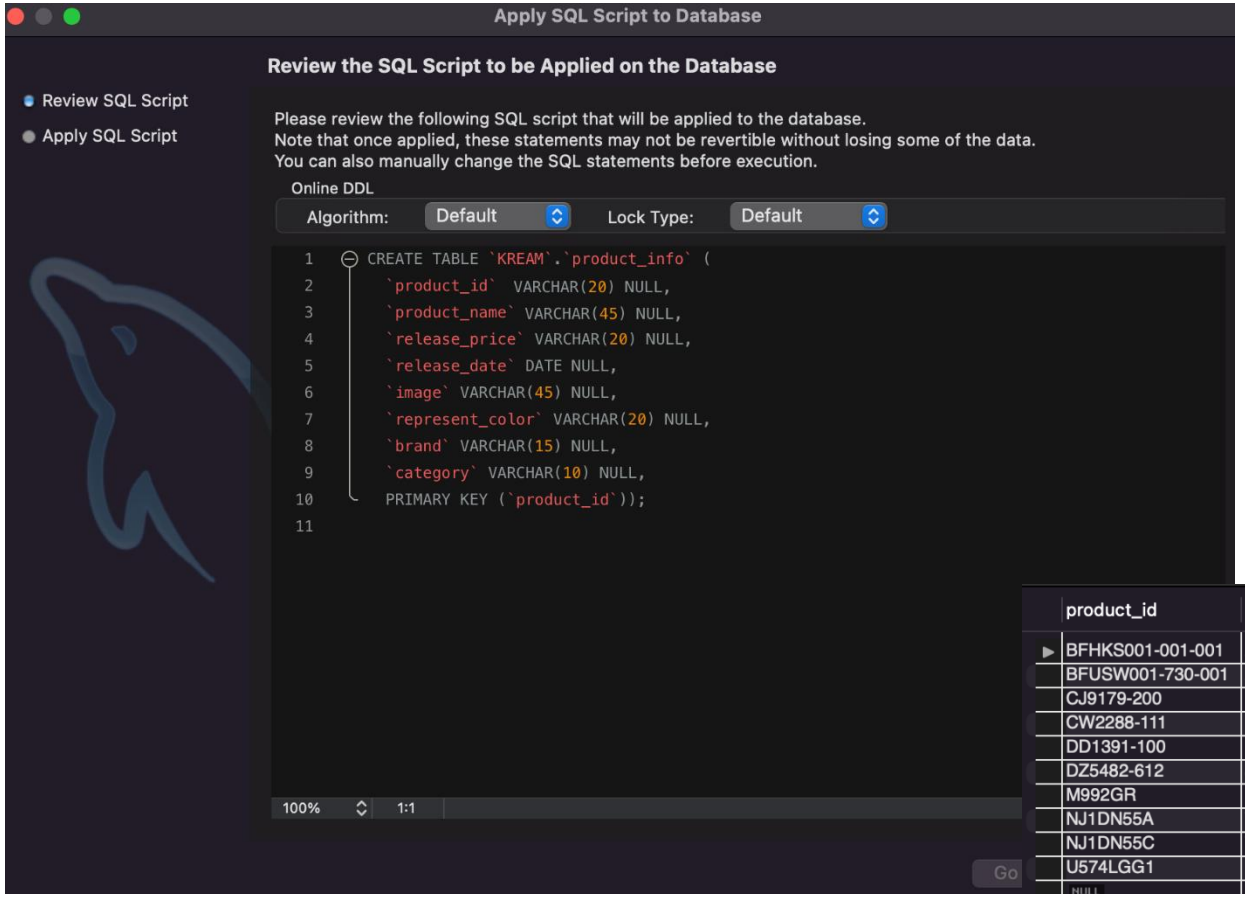




id	nickname	email	password	h.p	point	address	introduction	image	membership
1	dprlive	bgbg111@naver.com	12345	010-2222-1111	1000	인천시 부평구 산곡동 221	안녕하세요	image link	bronze
2	realisshoman	bgbg311@naver.com	12345	010-1111-2222	1000	인천시 부평구 산곡동 221	안녕하세요	image link	bronze
3	vata500	vata500@naver.com	12444445	010-1412-2325	1000	인천시 부평구 안녕동 10312-3	안녕하세요 접니다	image link	bronze
4	ovan	ovan00@naver.com	12444445	010-1441-3315	0	인천시 부평구 하이동 102-3 2동 1호	안녕하세요 접니다	image link	bronze
5	big_naughty	onaughty@naver.com	124124445	010-1312-4515	0	인천시 부평구 킥킥동 122-3 3동 1호	안녕하세요 접니다 저에요	image link	bronze
6	theq	theq@naver.com	124124445	010-4412-4111	100000	인천시 부평구 킥킥동 122-3 3동 1호	안녕하세요 접니다 저에요	image link	diamond
7	dok2	dok2@naver.com	1412545	010-1512-4391	2000	인천시 부평구 사상동 552-3 3동 1호	안녕하세요 접니다 저에요	image link	bronze
8	choilb	choilb@naver.com	15012495	010-1249-4191	2000	서울시 마포구 성산1동 112-3 2동 111호	안녕하세요 접니다 저에요	image link	bronze
9	giriboy	giriboy@naver.com	15303495	010-5549-5121	11000	서울시 마포구 성산1동 552-3 3동 111호	안녕하세요 접니다 저에요	image link	gold
10	swings	swings@naver.com	15swing95	010-1030-1121	6000	서울시 마포구 홍대동 354-3 5동 1010호	안녕하세요 접니다 저에요	image link	silver
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

DB생성 및 데이터 적재

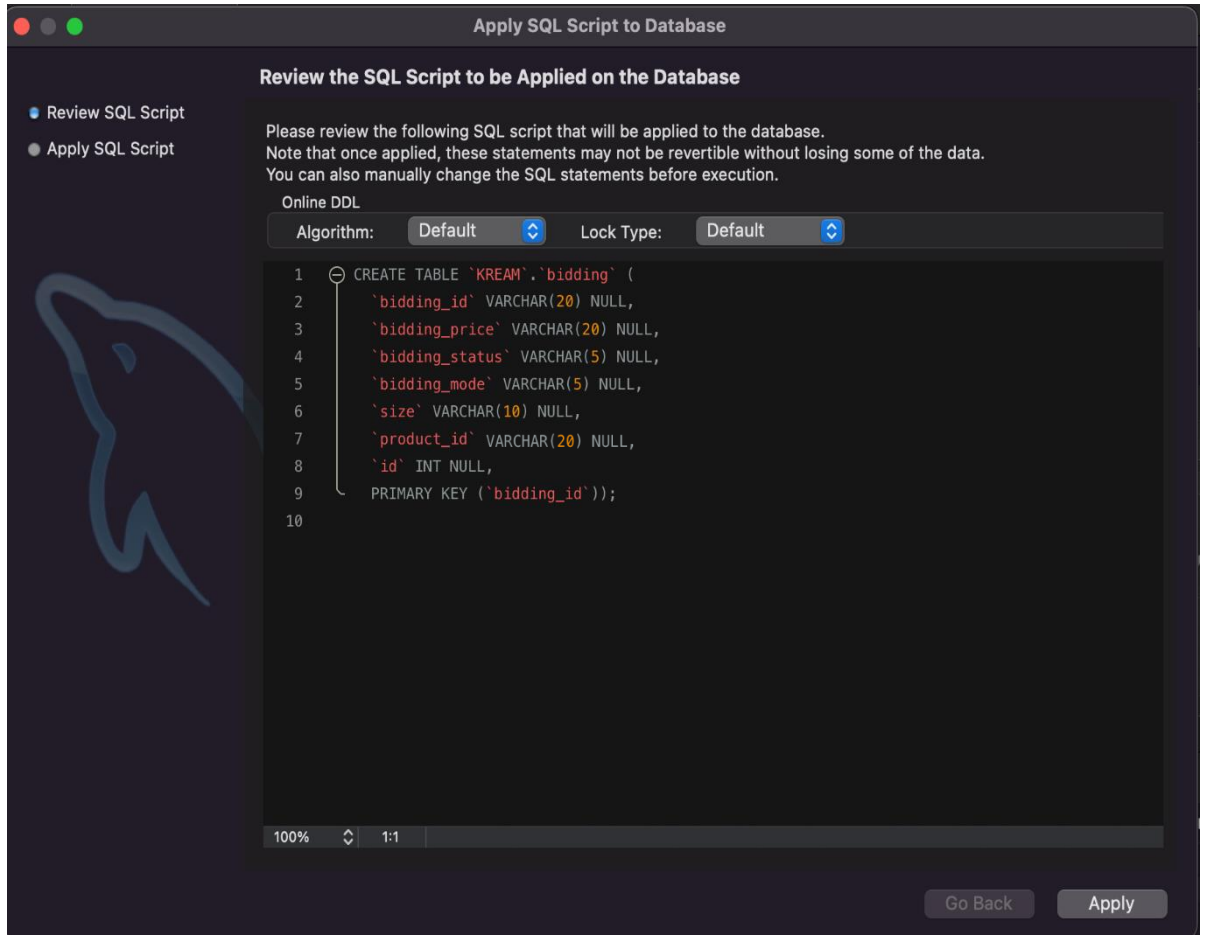
2. 상품 정보를 담은 product_info



product_id	product_name	release_price	release_date	image	represent_color	brand	category
BFHKS001-001-001	AMI de Coeur Wool Knit Black/Noir	485,000	NULL	image	Black/Noir	AMI	Clothes
BFUSW001-730-001	AMI de Coeur Sweatshirt Black - 22SS	NULL	NULL	image	Black	Ami	Clothes
CJ9179-200	Nike Air Force 1 '07 WB Flax	169,000	2019-09-10	image	FLAX/GUM/LIGHT/WHEAT	NIKE	Shoes
CW2288-111	Nike Air Force 1 '07 Low White	169,000	2019-09-10	image	WHITE	NIKE	Shoes
DD1391-100	Dunk Low Retro Black	129,000	2021-01-14	image	WHITE/BLACK	NIKE	Shoes
DZ5482-612	Jordan 1 Chicago	209000	2022-12-01	image	VASITY RED/BLACK	NIKE	Shoes
M992GR	New Balance 992 Made in USA Grey - D Standard	259,000	2020-04-13	image	GREY	New Balance	Shoes
NJ1DN55A	The North Face 1996 Eco Nuptse Jacket Black	339,000	NULL	image	Black	The North Face	Clothes
NJ1DN55C	The North Face 1996 Eco Nuptse Jacket White	339,000	NULL	image	Black/White	The North Face	Clothes
U574LGG1	New Balance 574 Legacy Black	129,000	2022-06-10	image	Black	New Balance	Shoes

DB생성 및 데이터 적재

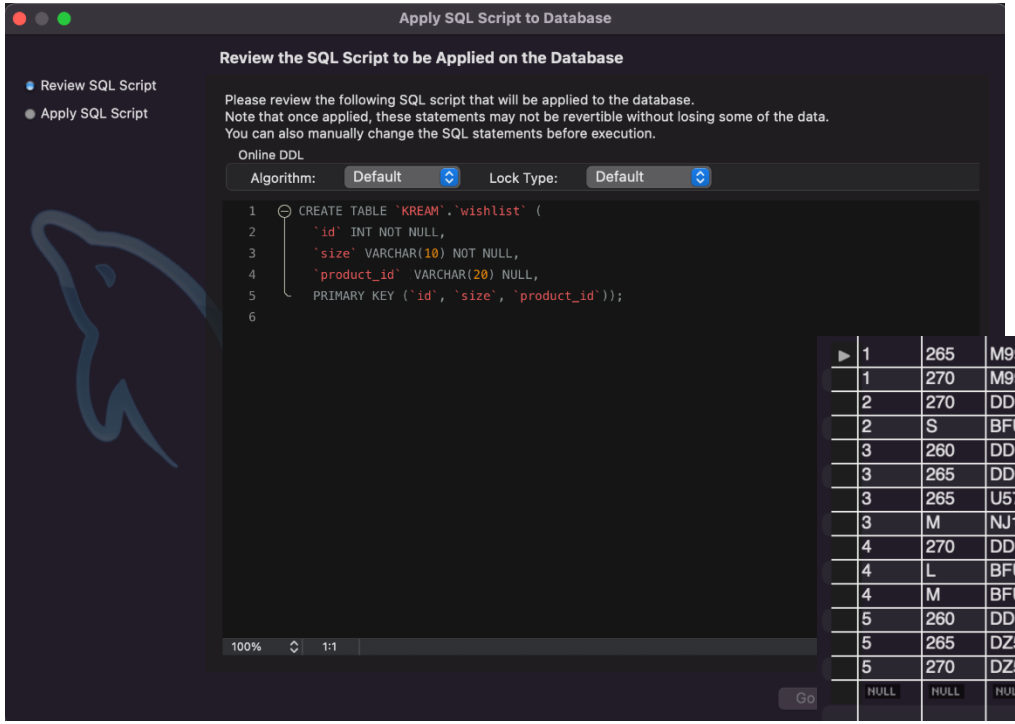
3. 입찰 정보를 담은 bidding



bidding_id	bidding_pri...	bidding_stat...	bidding_mo...	size	product_id	id
bid-1	139,000	ing	buy	270	DD1391-100	1
bid-10	144,000	ing	sell	270	DD1391-100	9
bid-11	142,000	ing	buy	270	DD1391-100	4
bid-12	142,000	ing	buy	260	DD1391-100	3
bid-13	182,000	ing	buy	260	U574LGG1	3
bid-14	141,100	done	buy	270	DD1391-100	7
bid-15	141,100	done	buy	270	DD1391-100	8
bid-16	141,100	done	sell	270	DD1391-100	6
bid-17	141,100	done	sell	270	DD1391-100	6
bid-18	181,000	ing	buy	270	U574LGG1	1
bid-19	301,000	ing	buy	270	DZ5482-612	1
bid-2	139,000	ing	buy	270	DD1391-100	2
bid-20	322,000	ing	buy	270	DZ5482-612	10
bid-21	333,000	ing	buy	270	DZ5482-612	1
bid-22	350,000	ing	buy	270	DZ5482-612	4
bid-23	360,000	ing	buy	270	DZ5482-612	5
bid-24	2790,000	ing	buy	270	NJ1DN55C	5
bid-25	279,000	ing	buy	270	NJ1DN55C	5
bid-26	279,000	ing	buy	270	NJ1DN55C	7
bid-27	310,000	ing	sell	270	NJ1DN55C	3
bid-29	310,000	ing	sell	270	BFUSW00...	5
bid-3	139,000	ing	buy	270	DD1391-100	1
bid-30	310,000	ing	sell	270	BFUSW00...	2
bid-31	310,000	ing	sell	270	BFUSW00...	4
bid-32	310,000	ing	sell	265	BFUSW00...	4
bid-33	310,000	ing	buy	265	BFUSW00...	1
bid-34	310,000	ing	buy	265	BFUSW00...	10
bid-35	310,000	ing	buy	265	NJ1DN55C	10
bid-36	310,000	ing	buy	S	NJ1DN55C	10
bid-37	310,000	ing	buy	M	NJ1DN55C	9
bid-38	320,000	ing	buy	M	NJ1DN55C	9
bid-4	143,000	ing	buy	270	DD1391-100	4
bid-5	143,000	ing	buy	270	DD1391-100	5
bid-6	144,000	ing	buy	270	DD1391-100	5
bid-7	144,000	ing	sell	260	DD1391-100	5
bid-8	144,000	ing	sell	260	DD1391-100	5
bid-9	144,000	ing	sell	270	DD1391-100	8
NULL	NULL	NULL	NULL	NULL	NULL	NULL

DB생성 및 데이터 적재

4. 계정 별 wishlist 정보를 담은 bidding



▶	1	265	M992GR
	1	270	M992GR
	2	270	DD1391-100
	2	S	BFUSW001-730-001
	3	260	DD1391-100
	3	265	DD1391-100
	3	265	U574LGG1
	3	M	NJ1DN55C
	4	270	DD1391-100
	4	L	BFUSW001-730-001
	4	M	BFUSW001-730-001
	5	260	DD1391-100
	5	265	DZ5482-612
	5	270	DZ5482-612
	NULL	NULL	NULL

(6) JDBC/MySQL 프로그램

응용의 기능 2개 구현

JDBC/MySQL 프로그램 - main function

메인 프로그램은 사용자 입력에 따라

1. 원하는 닉네임의 계정정보와 wishlist를 제공
2. 원하는 카테고리 내에서 가장 많이 거래된 제품의 순서대로 정렬 및 상품에 관한 정보를 제공합니다.
3. 프로그램을 종료합니다.

```
    public static void main(String[] args) throws ClassNotFoundException, SQLException{
        while(true) {
            Scanner scanner = new Scanner(System.in);
            System.out.println("-- 안녕하세요 한정판 거래 플랫폼 KREAM입니다 ---");
            System.out.println("-- 원하는 기능을 선택해주세요 ---");
            System.out.println("-- 1. 원하는 닉네임의 계정정보와 wishlist를 제공 ---");
            System.out.println("-- 2. 원하는 카테고리 내에서 가장 많이 거래된 제품의 순서대로 정렬 및 상품에 관한 정보 제공 ---");
            System.out.println("-- 3. 프로그램 종료 ---");
            String input = scanner.next();

            if(input.equals("1")) {
                searchuserinfo(scanner);
            }
            if(input.equals("2")) {
                sortbytrade(scanner);
            }
            if(input.equals("3")){
                System.out.println("프로그램을 종료합니다.");
                break;
            }
        }

    }
}
```

JDBC/MySQL 프로그램 - searchuserinfo

input 닉네임의 계정정보와 wishlist를 제공합니다.

```java
public static void searchuserinfo(Scanner scanner){
    Connection conn = null;
    PreparedStatement pstmt = null;
    PreparedStatement pstmt2 = null;
    ResultSet rs = null;
    ResultSet rt = null;
    System.out.println("닉네임 입력 시 닉네임에 관한 계정 정보와 wishlist 목록을 제공해드립니다.");
    System.out.println("닉네임을 입력해주세요!");
    String Name = scanner.next();

    try {
        conn = DriverManager.getConnection(DB_URL,USER,PASS);
        String query = "select nickname, email, point, address, introduction, image, membership\n"
                + "from account_info\n"
                + "where nickname = (?)";
        String query2 = "select size, product_id\n"
                + "from wishlist\n"
                + "where id in (select id from account_info where nickname = (?))";
        pstmt = conn.prepareStatement(query);
        pstmt.setString(1,Name);
        rs = pstmt.executeQuery();
        pstmt2 = conn.prepareStatement(query2);
        pstmt2.setString(1,Name);
        rt = pstmt2.executeQuery();
        while(rs.next()) {
            System.out.println(rs.getString("email") + " "+ rs.getInt("point") + rs.getString("membership"));
        }

        while(rt.next()) {
            System.out.println(rt.getString("size") + rt.getString("product_id"));
        }
        }catch (SQLException ex) {
            System.out.println("SQLException" + ex);
        }
}
```

```
-- 안녕하세요 한정판 거래 플랫폼 KREAM입니다 ---
-- 원하는 기능을 선택해주세요 ---
-- 1. 원하는 닉네임의 계정정보와 wishlist를 제공 ---
-- 2. 원하는 카테고리 내에서 가장 많이 거래된 제품의 순서대로 정렬 및 상품에 관한 정보 제공 ---
-- 3. 프로그램 종료 ---
1
닉네임 입력 시 닉네임에 관한 계정 정보와 wishlist 목록을 제공해드립니다.
닉네임을 입력해주세요!
ovan
ovan00@naver.com 0 bronze
270 DD1391-100
L BFUSW001-730-001
M BFUSW001-730-001
```

```
-- 안녕하세요 한정판 거래 플랫폼 KREAM입니다 ---
-- 원하는 기능을 선택해주세요 ---
-- 1. 원하는 닉네임의 계정정보와 wishlist를 제공 ---
-- 2. 원하는 카테고리 내에서 가장 많이 거래된 제품의 순서대로 정렬 및 상품에 관한 정보 제공
-- 3. 프로그램 종료 ---
1
닉네임 입력 시 닉네임에 관한 계정 정보와 wishlist 목록을 제공해드립니다.
닉네임을 입력해주세요!
dprlive
bgbg111@naver.com 1000 bronze
265 M992GR
270 M992GR
```

input 닉네임의 계정정보와 wishlist를 제공합니다.
이 때 sql query문을 두개 사용하여 두 개의 result set의 결과를 출력해줍니다.

JDBC/MySQL 프로그램 - sortbytrade

원하는 카테고리 내에서 가장 많이 거래된 제품의 순서대로 정렬 및 상품에 관한 정보를 제공합니다.

```
public static void sortbytrade(Scanner scanner) {
    Connection conn = null;
    PreparedStatement pstmt = null;
    ResultSet rs = null;
    System.out.println("원하시는 카테고리의 인기순 정렬 및 상품 정보를 제공해드립니다.");
    System.out.println("category를 입력해주세요 -> Shoes, clothes .. ");
    String a = scanner.next();
    try {
    conn = DriverManager.getConnection(DB_URL,USER,PASS);
    String query = "select product_name, brand, count(*) as cnt , image\n"
            + "from product_info as A , bidding as B\n"
            + "where A.product_id = B.product_id and A.category = (?) \n"
            + "group by A.product_id\n"
            + "order by count(*) DESC";
    pstmt = conn.prepareStatement(query);
    pstmt.setString(1,a);
    rs = pstmt.executeQuery();
    int Num = 1;
    while(rs.next()) {
        System.out.println(rs.getString("product_name") + " "+ rs.getString("brand") +" 거래개수 : " +  rs.getString("cnt") + " 제품 이미지 : " + rs.getString("image"));
        Num++;
    }
    conn.close();
    }
    catch (SQLException ex) {
        System.out.println("SQLException" + ex);
    }
}
```

```
-- 안녕하세요 한정판 거래 플랫폼 KREAM입니다 ---
-- 원하는 기능을 선택해주세요 ---
-- 1. 원하는 닉네임의 계정정보와 wishlist를 제공 ---
-- 2. 원하는 카테고리 내에서 가장 많이 거래된 제품의 순서대로 정렬 및 상품에 관한 정보 제공 ---
-- 3. 프로그램 종료 ---
2
원하시는 카테고리의 인기순 정렬 및 상품 정보를 제공해드립니다.
category를 입력해주세요 -> Shoes, clothes ..
shoes
Dunk Low Retro Black NIKE 거래개수 : 16 제품 이미지 : image
Jordan 1 Chicago NIKE 거래개수 : 5 제품 이미지 : image
New Balance 574 Legacy Black New Balance 거래개수 : 2 제품 이미지 : image
```

```
-- 안녕하세요 한정판 거래 플랫폼 KREAM입니다 ---
-- 원하는 기능을 선택해주세요 ---
-- 1. 원하는 닉네임의 계정정보와 wishlist를 제공 ---
-- 2. 원하는 카테고리 내에서 가장 많이 거래된 제품의 순서대로 정렬 및 상품에 관한 정보 제공 ---
-- 3. 프로그램 종료 ---
2
원하시는 카테고리의 인기순 정렬 및 상품 정보를 제공해드립니다.
category를 입력해주세요 -> Shoes, clothes ..
clothes
The North Face 1996 Eco Nuptse Jacket White The North Face 거래개수 : 8 제품 이미지 : image
AMI de Coeur Sweatshirt Black - 22SS Ami 거래개수 : 6 제품 이미지 : image
```

해당 카테고리의 거래 내역이 있는 제품의 거래 내역 개수와 제품의 이름, 브랜드가 차례로 정렬되어 출력됩니다.

(7) BCNF 정규화 및 스키마 정제

BCNF 정규화 및 스키마 정제

size(size)

category(category)

brand(brand)

bidding_status(bidding_status)

bidding_mode(bidding_mode)

membership(membership)

follow(follower_id, following_id)

like(id, post_id)

size_product(size, product_id)

wishlist(id, size_id, product_id)

product_tag(post_id, product_id)

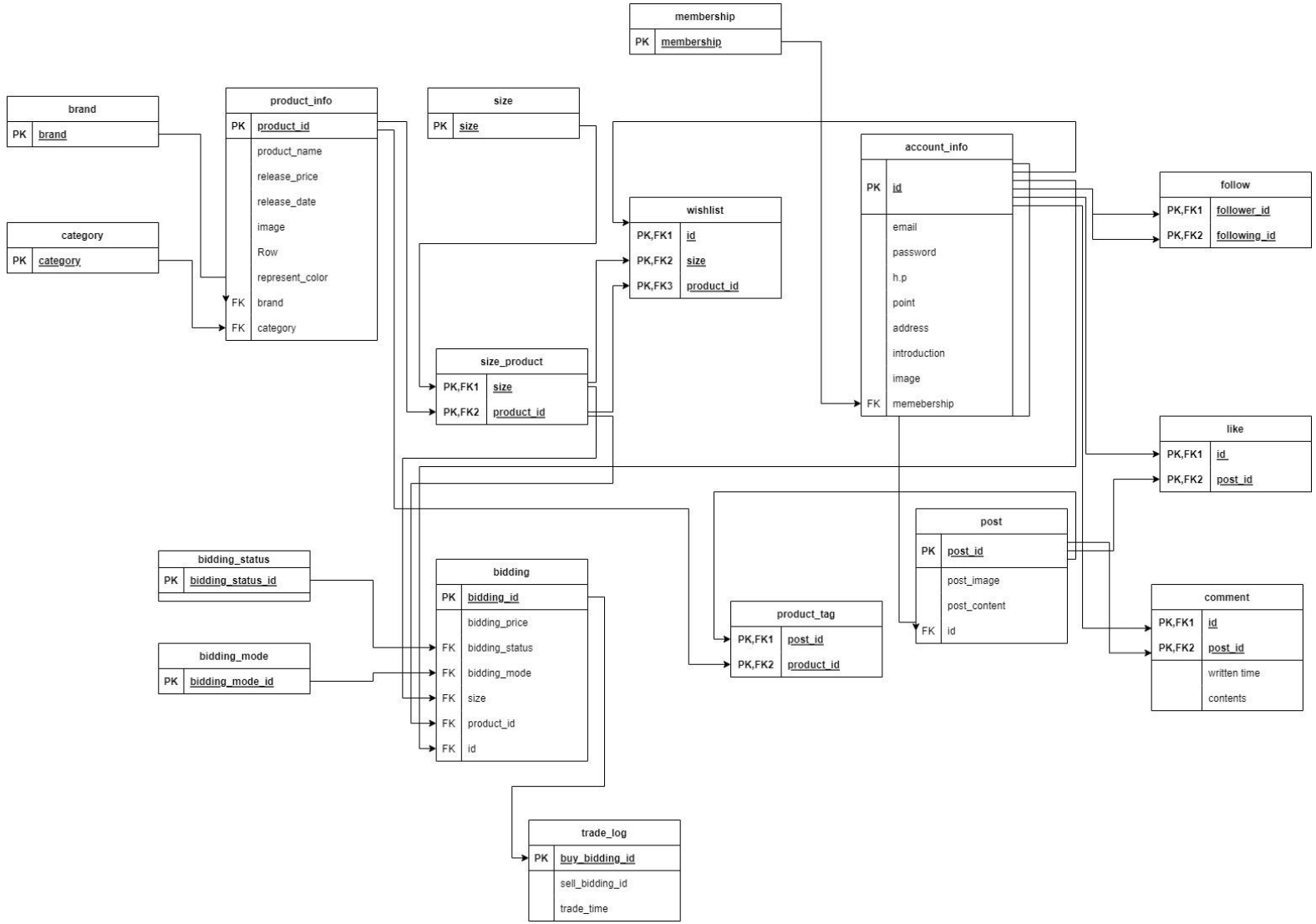
FD : nontrivial 함수 종속 없음

BCNF 정규화 및 스키마 정제

product_info(product_id, product_name, release_price, release_date, image, represent_color, brand, category, nickname)
FD : product_id -> product_name, release_price, release_date, image, represent_color, brand, category, nickname
//product_id 는 superkey
account_info(id, email, password, h.p, point, address, introduction, image, membership)
FD : id -> email, password, h.p, point, address, introduction, image, membership
//id는 superkey
bidding(bidding_id, bidding_price,bidding_status_id,bidding_mode-id,size_id,product_id,id)
FD : bidding_id -> bidding_price,bidding_status_id,bidding_mode-id,size_id,product_id,id
//bidding_id는 superkey
Trade_log(buy_bidding_id, sell_bidding_id, trade_time)
FD : buy_bidding_id -> sell_bidding_id, trade_time
//buy_bidding_id는 superkey
post(post_id, post_image,post_content,id)
FD : post_id -> post_image, post_content, id
//post_id는 superkey
comment(id, post_id,written_time, contents)
FD : id, post_id -> written_time, contents
// <id, post_id>는 superkey

BCNF 정규화 및 스키마 정제

BCNF 정규화 과정을 거친 후 특별하게 정제가 필요한 스키마가 없다고 판단하여 최종스키마는 그대로 유지



감사합니다.

중앙대학교 소프트웨어학부
20193802 조명근