Database System: Project 1(E-R design and Relational Schema design)

전공: 컴퓨터공학 학년: 3학년 학번: 20201635 이름: 전찬

**0. 목차**

1. 구현 목표

2. E-R diagram과 설계 설명

3. Relation schema와 설계 설명

4. SQL을 통한 Query statement

**1. 구현 목표**

이번 프로젝트는 온라인 / 오프라인에서 전자 제품을 판매하는 회사의 database를 어떻게 구현할지 구상하는 구현 목표를 갖는다. 이러한 회사의 예시로는, 우리 나라의 하이마트, 그리고 외국의 Circuit city 등이 존재한다. 이를 위해 E-R model과 E-R diagram, Relation schema, 그리고 구현한 Relation schema를 토대로 특정한 query를 수행할 수 있는 Query 문을 작성해 본다.

**2. E-R diagram과 설계 설명**

이번 프로젝트에서 기본적으로 존재해야 할 entity set은 다음과 같다.

1. product : 각 상품의 정보를 갖고 있는 entity set 이다.

product id, name, type, manufacturer, price 을 attribute로 갖는다.

2. customer : 고객의 정보를 갖고 있는 entity set, 주로 온라인 구매 고객의 정보에 관련해서 사용되지만, 포인트 적립 등 오프라인 고객에 관련해서도 사용할 수 있다.

customer id, name, age, address, number, card number 을 attribute로 갖는다.

3. store : 각 오프라인 매장의 정보를 저장하는 entity set 이다.

store id, address 을 attribute로 갖는다.

4. warehouse : 각 물류 창고의 정보를 저장하는 entity set 이다. 온라인 구매의 배송을 보내는 역할도 수행하며, store에서 상품의 재고가 부족하면, 이를 충당하는 역할도 수행한다.

warehouse id, address 을 attribute로 갖는다.

5. shipper : 배송하는 사람(배달 기사)의 정보이다.

shipper id, company name, location 을 attribute로 갖는다.

6. shipping : 배송되고 있는 택배에 대한 정보이다.

shipping id, destination, location, promised time 을 attribute로 갖는다.

7. sales : 상품을 판매할 때, 온라인 / 오프라인 판매 내역을 저장하는 entity set 이다.

sales id, date, total cost, location 을 attribute로 갖는다.

8. manufacturer : 한 상품의 제조사이다. warehouse의 재고가 부족한 경우 재고를 채워주거나, 구매자가 사용 중 해당 물품에 고장이 난 경우 수리해주는 역할을 수행한다.

company name, address 을 attribute로 갖는다.

9. package : 여러 product의 묶음 data 이다. 예를 들어, 마트에서 과자 묶음, 혹은 컴퓨터 세트 등이 존재할 수 있다.

product의 specialization 이며, multi value product id을 attribute로 갖는다.

relationship set은 다음과 같다.

1. purchase : product, customer, store 사이의 관계를 나타낸다.

2. contain : product와 다른 entity set 사이의 관계이다. store, warehouse, 혹은 구매한 내역에서 각 물품이 몇 개씩 존재하는지 파악할 수 있다.

3. stock : warehouse, store 사이의 관계로, 남아있는 재고를 의미한다.

4. supply : warehouse, manufacturer 사이의 관계로, 재고가 존재하지 않는 경우, 추가적인 공금을 받기 위한 relationship set 이다.

5. offline / online : 구매할 때 각 sales가 offline 구매인지, online 구매인지를 파악할 수 있는 relationship set 이다.

6. contract info : 계약이 되어있어서, 매달 돈을 지불하는 customer을 의미한다.

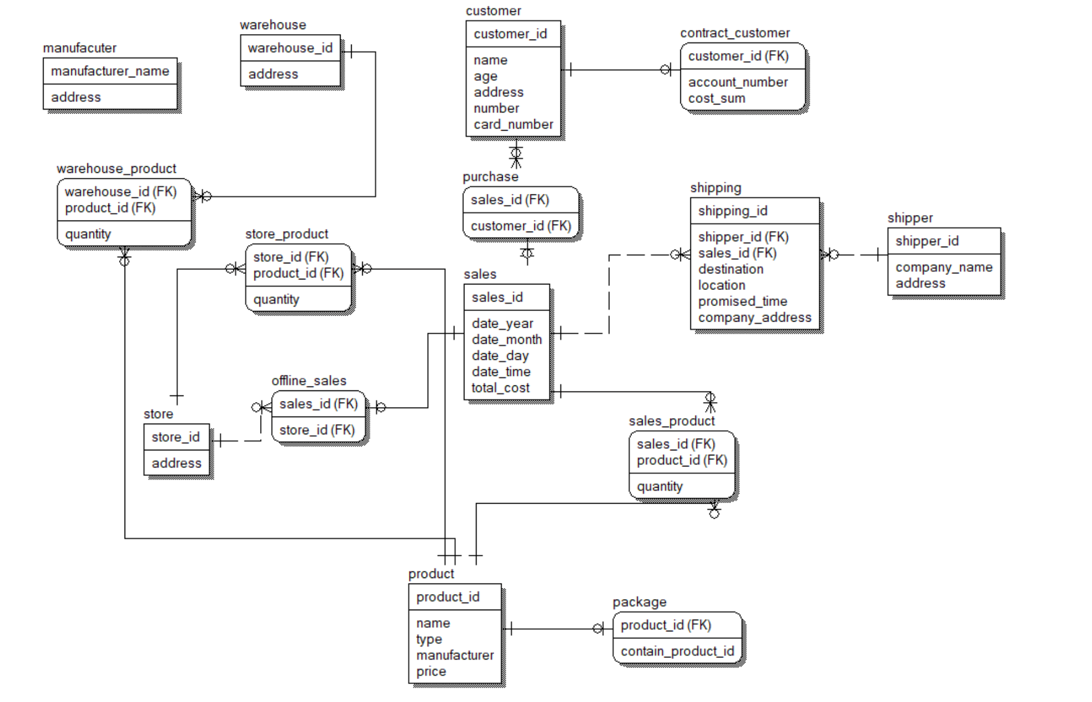
7. deliver : shipper에서 각 배송원(택배 기사님)이 존재하는데, 배송원이 어떤 물품을 배송하고 있는지를 파악할 수 있는 relationship set 이다.

따라서 이를 토대로 구현한 E-R diagram은 아래와 같다.



**2. Relation schema와 설계 설명**

relation schema는 위에서 구현한 E-R diagram을 바탕으로 만들어낼 수 있다. 변환할 때 여러 방법이 존재하는데, 예를 들어 relationship set의 경우 mapping cardinality에 따라서 primary key을 정하며, 이를 하나의 relation schema로 취급할 수 있다. 혹은 weak entity set의 경우, identifying entity set의 primary key을 추가해 줌으로써 하나의 relation schema을 구성할 수 있다. 이를 바탕으로 구현한 Relation schema는 아래와 같다.



위의 Relation schema를 보면, 위에서 설명한 entity set인 manufacturer, warehouse, store, customer, sales, product, package, shipping, shipper 이 존재하는 것을 파악할 수 있다. 또한 위의 E-R diagram과 다르게, shipping을 보면 shipper\_id, sales\_id 등이 추가된 것을 파악할 수 있는데, 이는 이전에 존재하던 online / deliver relationship set을 새로운 relation으로 만들 때 생길 수 있는 redundancy를 줄이기 위해, relationship set을 relation으로 만들지 않고, Foreign key 등을 추가하며 위와 같이 설정했다. 또한 store\_product, sales\_product 등과 같이 많이 사용할 수 있는 relationship set은 새로운 relation으로 정의한 형태임을 파악할 수 있다.

각 relation과 attribute의 조건에 대한 특징은 아래와 같다.

1. manufacter : 어떠한 product를 만드는 회사이다.

2. warehouse : 물류 창고를 의미한다. 이때 이 relation 의 attribute quantity >= 0 이 항상 성립해야 한다.

3. store : 각 오프라인 가게를 의미한다.

4. product : 상품을 의미한다.

5. customer : 고객을 의미한다. 이때 age, card\_number, address 등은 null 고객이 입력하지 않으면 null이 될 수도 있다.

6. purchase : 각 sales에 대해 어떤 고객이 구매했는지를 알려주는 relation 이다.

7. sales : 각 sales 이다. 판매 가격과 함께 판매한 날짜에 대한 attributes 를 갖는다.

8. shipping : 배송을 의미한다. destination인 목적지, location인 현재 위치 등을 attributes로 갖는다.

9. shipper : 배송하는 기사를 의미한다. 배송하는 기사의 현재 위치인 address가 존재한다.

10. package : 여러 product에 대한 묶음 상품이다.

11. warehouse\_product : 각 물류 창고에 어떤 물품이 얼마나 존재하는지 파악하는 값이다. 이때 quantity >= 이 되어야 한다.

12. store\_product : 각 오프라인 매장에 어떤 물품이 얼마나 존재하는지 파악하는 값이다. 이때 quantity >= 이 되어야 한다.

13. offline\_sales : 각 sales에 대해서 offline에서 판매한 경우, 그 store가 어디인지 알 수 있는 relation 이다.

14. contract\_customer : 계약이 있는 고객을 의미한다. 계약이 있는 고객의 경우, cost\_sum을 매달 말에 지불하며, 지불한 경우 cost\_sum 은 다시 0이 되는 형태이다. 또한 cost\_sum >= 0 이다.

15. sales\_product : 각 판매에서 어떤 물품이 얼마나 판매되었는지를 파악하는 값이다. 이때 quantity >= 이 되어야 한다.

**4. SQL을 통한 Query statement**

- Assume the package shipped by USPS with tracking number 123456 is reported to have been destroyed in an accident. Find the contact information for the customer. Also, find the contents of that shipment and create a new shipment of replacement items.

select number

from customer natural join purchase natural join sales natural join shipping

where shipping\_id = 123456

- Find the customer who has bought the most (by price) in the past year.

select customer\_id, customer\_name

from customer join purchase on customer\_id join sales on sales\_id

group by customer

having sum(total\_cost) = max(sum(total\_cost))

- Find the top 2 products by dollar-amount sold in the past year.

select TOP 2 product\_id

from select product\_id, sum(quantity \* price) as total\_amount

from sales\_product natural join product

group by product\_id

order by total\_amount desc

- Find the top 2 products by unit sales in the past year.

select TOP 2 product\_id

from select product\_id, count(quantity) as total\_count

from sales\_product natural join product

group by product\_id

order by total\_count desc

- Find those products that are out-of-stock at every store in California.

select product\_id

from store\_product join store

where quantity = 0 and address = ‘California’

- Find those packages that were not delivered within the promised time.

select product\_id

from shipping natural join sales natural join sales\_product\_natural join package

where today\_time > promised\_time

- Generate the bill for each customer for the past month.

select customer\_id, sum(total\_cost) as bill

from customer natural join purchase natural join sales

where date\_month = today\_month – 1 or

today\_month = 1 and date\_year = today\_year – 1 and date\_month = 12