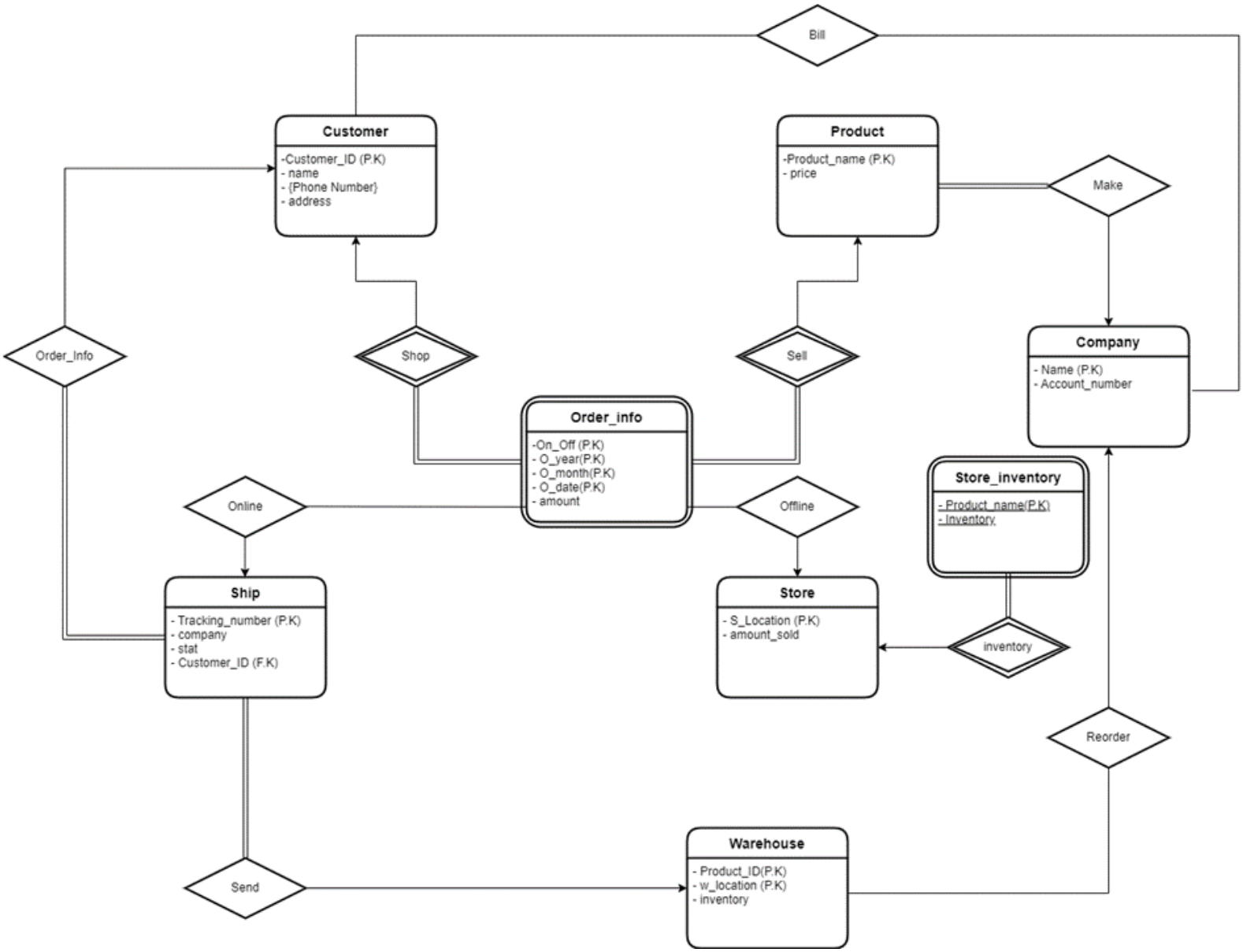
데이터베이스시스템(CSE4110) Project 2

20181632 박성현

1. E-R model



몇가지 사항을 수정했다. Customer에 dollar_used, Product에서 Product_ID를 Producr_name으로, amount_sold, dollar_sold, type를 제거했다. 또한 Time relation을 제거하고 Order_info에 구매 날짜를 P.K로 추가했다. Order_info에는 구매시 구매한 개수를 뜻하는 amount로 추가했다. 또한 send와 reorder relationship을 many to many 에서 one to many로 변경했다. Send의 경우 하나의 배송은 하나의 warehouse에서밖에 물품을 못 받기 때문이고 reorder의 경우 warehouse의 각 tuple은 하나의 창고에서의 하나의 물품을 뜻하기에

그 물품은 하나의 회사에서 밖에 오지 못한다. Buy relationship의 경우 order_info와 중복이기에 삭제시켰다.

2. BCNF with testing

BCNF를 고려했을 때, 바꾼 점은 없었다. 모두 BCNF를 만족시킨다. 예를 들어 Ship이 BCNF인지 확인해보자. BCNF를 만족시키려면 존재하는 functional dependency들이 trivial하거나 a->b에서 a가 superkey여야 한다. ship에서 S_company나 stat, Customer_ID, W_location은 모두 중복이 가능하다. 이 속성만으로 하나의 tuple을 결정할 수 없다. Primary key인 Tracking number은 unique하기에 다른 모든 속성을 define할 수 있다. 그러므로 tracking number->S_company, stat, Customer_ID, W_locaiton의 functional dependency가 존재한다. 여기서 tracking number은 superkey이기에 BCNF를 만족한다. 수정된 entity들과 relationship들은 아래와 같다.

Customer(Customer_ID, C_name, address)

Product(Product_name, price, Company_Name (F.K))

Order_info(On_Off, Product_name, (F.K) Customer_ID (F.K), O_year, O_month, O_date , Tracking_Number (F.K), S_Location (F.K), amount)

Store(S_Location, amount_sold)

Store_inventory(Product_name, S_Location (F.K), inventory)

Warehouse(Product_name, w_location, inventory, Name(F.K))

Company(Company_Name, Account_number)

Ship(Tracking_Number, S_company, stat, Customer_ID (F.K), W_location(F.K))

Bill(Customer_ID, Company_Name)

Replenish(Product_name, S_Location, w_location)

Send(Tracking_number, Product_name, w_location)

Customer_Phone(Customer_ID (F.K), Phone Number)

3. Query

Company

Company_Name	Account
Apple	1000011705
Bannan	1000021447
Orange	1000032673
Grape	1000025664
Melon	1000015749
Onestar	1000007509
Twostar	1000003707
Fourstar	1000020216
Fivestar	1000016547
Sixstar	1000017770
Onemoon	1000004145
Twomoon	1000012602
Fourmoon	1000025589
Fivemoon	1000031854
Sixmoon	1000002673

Customer

Customer_ID	Name	Address
13762	Kim	Yantai
31624	Lee	New York
77514	Park	Busan
19012	Song	Paris
65982	Woo	Beijing
86411	Jang	Hawaii
9463	Hong	California
17328	Sally	Incheon
34312	Tom	Las vagas
53297	Kelvin	California
82379	Jack	LA
98674	Jimmy	North Korea
70831	Paul	Busan
14709	Eric	Yantai
38806	Hello	Seoul

Order_info

On_Off	Product_name	Customer_ID	O_year	O_month	O_date	Tracking_number	S_location	amount
On	Camera	34312	2022	01	04	11111	null	4
On	Computer	77514	2021	07	12	22222	null	3
On	Microwave	31624	2022	02	15	33333	null	2
On	TV	13762	2021	04	27	44444	null	1
On	Phone	19012	2021	12	01	55555	null	2
On	AC	70831	2022	05	30	66666	null	5
On	Audio	38806	2022	05	24	77777	null	3
On	Camera	86411	2021	01	12	88888	null	2
On	Clock	77514	2021	11	13	99999	null	5
On	Watch	31624	2022	05	15	10100	null	10
On	Phone	14709	2022	04	07	11110	null	2
On	TV	34312	2021	07	06	12120	null	1
On	Camera	65982	2021	05	08	13130	null	4
On	Washer	17328	2022	03	22	14140	null	1
On	Controller	19012	2021	09	25	15150	null	1
Off	Computer	53297	2021	02	24	null	California	2
Off	Mouse	82379	2022	05	22	null	LA	5
Off	TV	82379	2021	09	17	null	LA	1
Off	AC	38806	2021	07	11	null	Seoul	1
Off	AC	9493	2022	05	03	null	California	1

Product

Product_name	Company_name	price($)
Camera	Apple	100
Computer	Apple	500
Keyboard	Orange	20
Phone	Apple	400
Mouse	Grape	10
TV	Twostar	700
Microwave	Twomoon	50
AC	Sixstar	300
Heater	Onemoon	90
Battery	Fourstar	20
Audio	Fivemoon	150
Clock	Fourstar	20
Watch	Fourmoon	70
Washer	Melon	200
Controller	Sixmoon	250

Store

S_location	S_dollar_sold
California	1300
LA	1200
New York	0
Seoul	300
Busan	0
Incheon	0
North Korea	0
Beijing	0
Hawaii	0
Paris	0
Berlin	0
Tokyo	0
Yantai	0
Moscow	0
Las vagas	0

Ship

Tracking_number	S_company	stat	Customer_ID	Product_name	W_location
11111	A	Ontime	34312	Camera	New York
22222	B	Ontime	77514	Computer	Paris
33333	A	Late	31624	Microwave	New York
44444	C	Ontime	13762	TV	New York
55555	B	Late	19012	Phone	Seoul
66666	B	Destroyed	70831	AC	New York
77777	C	Ontime	38806	Audio	Seoul
88888	A	Late	86411	Camera	Paris
99999	B	Ontime	77514	Clock	Paris
10100	B	Ontime	31624	Watch	New York
11110	C	Ontime	14709	Phone	New York
12120	A	Late	34312	Seoul	Seoul
13130	C	Ontime	65982	Camera	Seoul
14140	C	Destroyed	17328	Washer	Paris
15150	B	Late	19012	Controller	New York

위의 테이블은 몇 개의 중요한 entity에 대한 정보이다.

기본적인 고객의 정보는 Customer, 물품의 정보는 Product에 저장되어 있다. Order_info 에는 구매내역에 대한 정보가 들어있다. 누가 어떤 물품을 언제 몇 개 온라인 혹은 오프라인으로 몇 개 샀는지에 대한 정보가 들어있다. Ship에는 온라인 구매로 이루어진 물품의 tracking number, 배송회사, stat, 구매자, 물품, 그리고 물품을 받은 창고의 위치가 저장되어 있다. 또한 store에는 상점의 위치와 총 판매금액이 나와있다. 구매이력에서 offline구매는 5개이므로 5개의 구매에 대한 판매 금액이 저장되어 있다. Store inventory라는 entity에는 총 물품 15개*상점 15개=225개의 tuple이 삽입되어 있다. 각 tuple은 물품의 이름, 상점의 위치, 그리고 재고를 저장하고 있다. 창고는 배송하는 물품을 보내주는 곳이다. warehouse라는 entity는 창고의 위치, 물품의 이름, 재고, 그리고 그 물품의 제조회사에 대한 정보를 담고 있다. 창고는 총 3개로 3*15, 즉 45개의 tuple을 가지고 있다.

Query 작성시에 create table과 insert data는 txt파일로 코드를 읽어와 query문을 넘겨주는 식으로 진행하였다. Create table과 insert data는 두개의 다른 txt파일, 20181632_create와 20181632_insert로 구성되어 있다.

1. **Assume the package shipped by USPS with tracking number X is reported to have been destroyed in an accident. Find the contact information for the customer.**

   "select T.Customer_ID, T.Phone_number from Customer_phone as T, Ship as S where S.stat= 'Destroyed' and S.Customer_ID = T.Customer_ID"

   다음과 같은 query문을 사용했다. Ship에서 stat이 destroyed된 tuple들을 가져와 고객의 번호가 있는 Customer_phone entity의 Customer_Id와 매치시켜 출력했다. 결과로 배송중 파손된 물품의 고객 ID, 그리고 전화번호가 출력된다.

   1-1. Then find the contents of that shipment and create a new shipment of replacement items

   "select Tracking_number,Product_name from Ship where Ship.stat= 'Destroyed'"

다음과 같은 query문을 사용해 ship에서 파손된 물품의 이름과 tracking number을 가져와 출력한다. 그 후, 업데이트 할 건지 옵션선택 후, 만약 업데이트를 하면

"update Ship set Tracking_number = Tracking_number+10 where stat = 'Destroyed'" 로 새로운 tracking number을 할당해 준다.

또한 "update Ship set stat='reshipping' where stat='Destroyed'" 를 사용해 stat을 reshipping, 즉 재배송이라고 설정한다. 마지막으로

"select * from Ship where stat='reshipping'"를 통해 update된

배송정보를 출력한다.

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

"select O.amount,P.price, C.C_name from Order_info as O, Product as P, Customer as C where O.Product_name=P.Product_name and O.O_year = '2021'and C.Customer_ID=O.Customer_ID"

위와 같은 query문을 사용했다. 먼저 주문 정보에서 현 시점 기준 작년인 2021년에 구매한 이력의 구매자 이름과 구매 물품 가격, 구매 개수, 물품이름을 가져왔다. 구매자의 이름을 name배열에 저장하고 used 배열에는 물품의 가격과 구매 개수를 곱해서 총 사용한 금액을 저장했다. 또한, 한 사람이 여러 번 구매를 했을 경우, 한 사람에 대해 두개의 index가 존재하므로 같은 값을 가진 name 배열에 대해 이름을 지우고 그 index의 used배열의 값을 서로 더해 중복을 제거했다. 중복되어 제거된 값은 name=X, used=0이다. 그리고 used의 max값을 찾아 출력했다.

Park이라는 사람이 computer 3개=1500+clock 5개=100=1600으로 2021년에 가장 많은 소비를 했다.

2-1. Then find the product that the customer bought the most.

"select O.amount,O.Product_name from Order_info as O where O.O_year = '2021'"

위의 query문을 사용했다. 2021년 구매이력에서 물품의 이름과 구매개수를 가져왔다. 2번과 같이 name배열에 물품의 이름, amount배열에는 구매개수를 저장했다. 물품의 이름의 중복이 있을 수 있으므로 중복을 제거하고 amount를 적절히 조절했다. 그 후, amount에서 max값을 찾아 출력했다. 2021년에는 카메라가 4+2개 총 6개로 가장 많이 팔렸다.

3. Find all products sold in the past year.

"select O.amount,P.price, P.Product_name from Order_info as O, Product as P where O.Product_name=P.Product_name and O.O_year = '2021'"

위의 query문을 사용했다. 2021년의 구매이력 중, 가격과 구매 개수와 물품이름을 가져와 출력했다2번과 같은 방식으로 name에는 물품의 이름, used에는 가격과 구매 개수를 곱해 총 가격을 계산했다. 가격에 대한 내림차순으로 정렬했다. 물품의 이름과 총 팔린 가격이 출력된다. 값을 출력하며 2021년에 팔린 총 물품의 개수 count도 계산해 준다.

3-1. Then find the top K products by dollar-amount sold.

k값을 입력 받고, k만큼 for loop을 돌면서 가격으로 정렬된 배열 name과 used를 출력한다. 여기서 X가 name의 값으로 들어가 있으면 중복을 제거한 값이므로 건너뛴다. max값은 count의 값이다.

3-2. And then find the top 10% products by dollar-amount sold.

위에서 계산한 count에 0.1을 곱해 10%의 값을 찾는다. Float형 이므로 int로 바꾸기 위해 1을 더해준다. 그리고 그 값만큼 for loop에서 name과 used를 출력한다.

4. Find all products by unit sales in the past year

"select O.amount, O.Product_name from Order_info as O where O.O_year = '2021'"

다음과 같은 query문을 사용해 2021년 구매이력에서 구매 개수, 물품이름을 가져와 출력했다. 위에서 사용한 방법과 똑같이 name 배열에 물품의 이름, amount에 물품의 구매 개수를 저장했고, 중복을 제거 후 정렬하여 물품의 이름과 구매 개수를 출력한다. 출력하며 물품의 개수 count를 계산한다.

4-1. Then find top K products by unit sales.

K값을 입력 받고 K 만큼 name과 amount배열을 출력한다. Name과 amount는 이미 amount에 대해 내림차순으로 정렬되어 있으므로 그대로 출력한다. Max값은 count이다.

4-2. And then find the top 10% products by units sales.

Count에 0.1을 곱해 그 값 만큼 name과 amount배열을 출력한다.

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

"select * from Store_inventory as S where S.S_location = 'California' and S.S_inventory = '0'"

위의 query문을 사용하여 store_inventory 에서 위치가 California이고 재고가 0인 tuple을 모두 가져와 출력한다. 물품의 이름, 상점의 위치, 그리고 재고가 순서대로 출력된다. Battery, Mouse, Washer의 재고가 0인 것을 알 수 있다.

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

Ship 에서 Ontime이외의 모든 stat은 제시간에 도착하지 못한 배송이다.

"select * from Ship as S where S.stat = 'Late'"

"select * from Ship as S where S.stat = 'Destroyed'"

"select * from Ship as S where S.stat = 'reshipping'"

세개의 query문으로 stat이 Late, Destroyed, reshipping인 모든 tuple들을 가져와 출력한다. 배송의 tracking number, 배송회사, stat, 고객_ID, 물품 이름이 순서대로 출력된다.

7. **Generate the bill for each customer for the past month.**

현 시간 기준 지난 달은 2022-05이다.

"select O.amount,P.price, C.C_name from Order_info as O, Product as P, Customer as C where O.Product_name=P.Product_name and O.O_year = '2022'and O.O_month = '5' and C.Customer_ID=O.Customer_ID"

가장 먼저 2022년 5월에 이루어진 구매에 대한 정보를 모두 가져온다. Name 배열에는 구매자의 이름, used배열에는 그 구매에서 사용한 총 금액을 저장한다. 중복을 제거해준다.

"select C.C_name, C.Customer_ID from Customer as C"

모든 고객에 대한 bill을 출력해야 하기 때문에 다음 query문으로 모든 고객의 정보를 가져온다. C_name에는 고객의 이름, C_ID에는 고객의 ID를 저장한다.

"select * from Order_info where O_year = '2022' and O_month = '5'"

마지막으로 2022년 5월에 이루어진 구매에 대한 정보를 다시 한번 더 가져와 저장한다. 첫번째 query와 나누어서 실행한 이유는 중복의 유무이다. 첫번째 query에서는 구매자의 이름과 총 사용금액을 가져와야 하기에 한 사람의 두번의 구매에 대해서 따로 나타나면 안된다. 그러므로 중복을 제거해주었다. 하지만 이번 query는 중복을 제거하지 않는다. 이는 Bill을 출력할 때, 각 구매에 대한 정보를 출력하기 위해서이다. 또한 첫번째의 query에서 select *로 모든 값을 가져오게 되면 Customer과 Product의 속성도 모두 가져오므로 데이터 공간의 낭비가 될 수 있다.

For loop을 통해 모든 고객에 대해 구매 이력을 출력한다. 출력할 구매 이력이 없을 경우, flag값을 설정하여 No Purchase in 2022-05문을 출력한다. 구매 이력이 있을 경우, 구매 이력에 대한 모든 정보, 즉 Order_info에서 가져온 모든 정보를 출력하고 마지막에 used배열에 있는 그 고객이 2022년 5월에 사용한 총 금액을 출력한다.