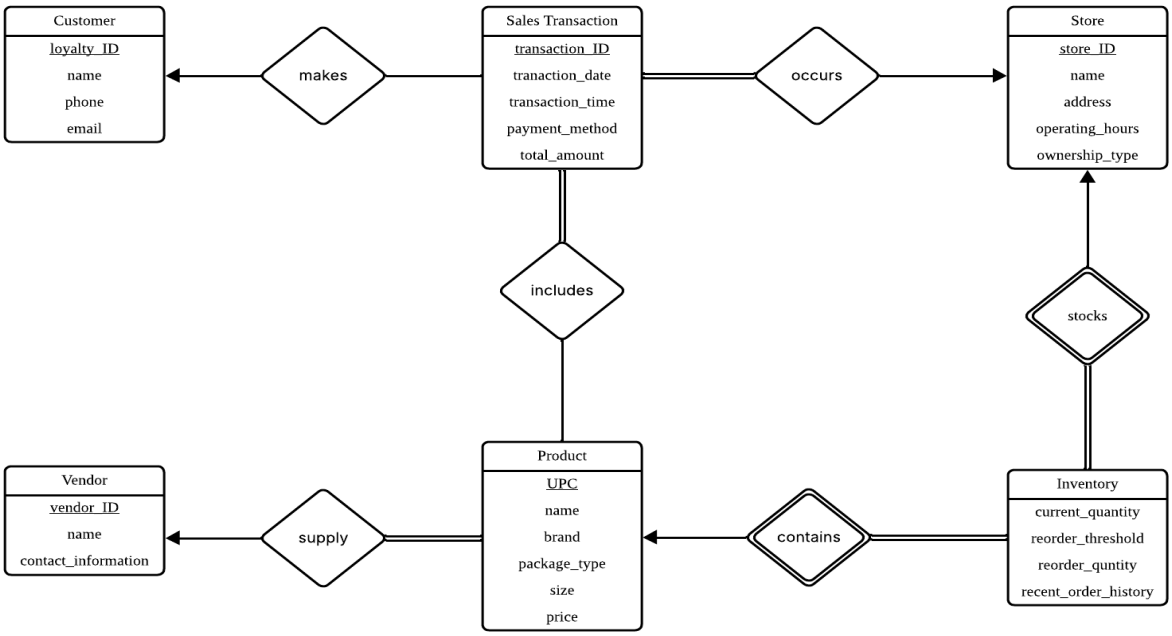
CSE4110 - Database System

DB Project 1 Report

20190808 방지혁

1. Final ER Diagram



2. Short Description about Entities and Relationships

Entities

Customer (소비자)

- (loyalty_id, name, phone, email)

- loyalty_id를 PK로 가지고, name, phone, email을 속성으로 가진다.

Sales_Transaction (판매 거래)

- (transaction_ID, transaction_date, transaction_time, payment_method, total_amount)

- transaction_ID를 PK로 가지고, transaction_date, transaction_time, payment_method, total_amount를 속성으로 가진다.

Store (매장)

- (store_ID, name, address, operating_hours, ownership_type)

- store_ID를 PK로 가지고, name, address, operating_hours, ownership_type을 속성으로 가진다.

Product (제품)

- (UPC, name, brand, package_type, size, price)

- UPC를 PK로 가지고, name, brand, package_type, size, price를 속성으로 가진다.

Vendor (공급업체)

- (vendor_ID, name, contact_information)

- vendor_ID를 PK로 가지고, name, contact_information을 속성으로 가진다.

Inventory (재고)

- (store_ID, UPC, current_quantity, reorder_threshold, reorder_quantity, recent_order_history)

- 약한 엔티티로 Store의 store_ID와 Product의 UPC를 합쳐서 복합 PK로 가지며, current_quantity, reorder_threshold, reorder_quantity, recent_order_history를 속성으로 가진다.

Relationship

Makes: (customer – sales_transcation 1 : M)

customer와 sales_transaction은 일대다 관계를 가진다. 한 구매자가 여러 구매를 할 수 있기 때문이다. 그리고, loyalty 등록이 된 customer 만 등록되어 있다. 그렇기에 다른 등록되어 있지 않은 일반 소비자들도 sales_transaction에 참여할 수 있고, 모든 loyalty customer가 sales_transaction에 참여하지 않을 수 있기 때문에, total participation 및 다른 complex constraint는 적용되지 않는다.

Occurs: (sales_transaction – store M : 1)

sales_transaction과 store는 다대일 관계를 가진다. 여러 판매 거래가 하나의 매장에서 발생하기 때문이다. 모든 sales_transaction은 반드시 어떤 store에서 발생해야 하므로, sales_transaction 측에서는 total participation이 적용된다. 그러나 아직 거래가 발생하지 않은 새로운 매장이 있을 수 있으므로, store 측에서는 total participation이 적용되지 않는다.

Includes: (sales_transaction – product M : N)

sales_transaction과 product는 다대다 관계를 가진다. 하나의 판매 거래는 여러 제품을 포함할 수 있고, 하나의 제품은 여러 판매 거래에 포함될 수 있기 때문이다. 모든 sales_transaction은 최소한 하나의 product를 포함해야 하므로, sales_transaction 측에서는 total participation이 적용된다. 그러나 아직 판매된 적이 없는 제품이 있을 수 있으므로, product 측에서는 total participation이 적용되지 않는다.

Supply: (vendor – product 1 : N)

vendor와 product는 일대다 관계를 가진다. 한 공급업체가 여러 제품을 공급할 수 있기 때문이다. 모든 product는 반드시 어떤 vendor에 의해 공급되어야 하므로, product 측에서는 total participation이 적용된다. 그러나 아직 어떤 제품도 공급하지 않는 새로운 공급업체가 있을 수 있으므로, vendor 측에서는 total participation이 적용되지 않는다.

Stocks: (store – inventory 1 : N)

store와 inventory는 일대다 관계를 가진다. 한 매장은 여러 제품의 재고를 가질 수 있기 때문이다. 모든 inventory 항목은 반드시 어떤 store에 속해야 하므로, inventory 측에서는 total participation이 적용된다. 또한, 이 관계는 약한 관계 집합인 inventory가 참여하는 관계이다.

Contains: (product – inventory 1 : N)

product와 inventory는 일대다 관계를 가진다. 한 제품은 여러 매장에 재고로 존재할 수 있기 때문이다. 모든 inventory 항목은 반드시 어떤 product에 대한 것이어야 하므로, inventory 측에서는 total participation이 적용된다. 그러나 아직 어떤 매장에도 재고로 존재하지 않는 제품이 있을 수 있으므로, product 측에서는 total participation이 적용되지 않는다.

3. Coverage of Sample Queries

3-1) 특정 제품 (UPC, 이름, 브랜드로 확인)을 현재 취급하는 매장과 그 재고량은?

Product entity와 Inventory entity를 연결하여 구한다. Product 엔티티에서 UPC, 이름, 혹은 브랜드로 해당 제품을 식별한 다음, contains 관계를 통해 해당 제품의 Inventory 정보를 찾을 수 있다. Relational schema로 전환할 시 store id가 inventory relation에 FK이자 PK로 들어오기 때문에 이를 통해 해당 제품을 보유한 Store 정보를 얻을 수 있으며 또한 Inventory의 current_quantity라는 속성이 있기 때문에 해당 제품 재고량을 확인할 수 있다.

3-2) 지난 달 각 매장에서 판매량이 가장 높은 제품은?

Relational schema로 변환 시 transaction_ID와 UPC를 복합 PK(Primary Key)로 가지는 Transaction_Product라는 새로운 테이블이 생성될 것이다. Transaction_Product 테이블과 Sales_Transaction 테이블을 조인하여 지난 달 거래 내역 추출하고, 매장별로 그룹화한다. 각 매장-제품 조합별로 count하여 계산하고, 내림차순 정렬하여 최고 판매 제품을 도출한다.

3-3) 이번 분기에 가장 높은 총 매출을 올린 매장은?"

Relational schema에서는 Sales_Transaction table이 store_ID 외래 키를 가질 것이다. 그렇기에 Sales_Transaction 테이블에서 특정 날짜 범위로 거래를 필터링하고, store_ID로 그룹화한 다음 total_amount의 합계를 계산한다. 그 후 내림차순 정렬하여 최고 매출 매장을 찾아낸다.

3-4) "가장 많은 제품을 공급하는 vendor와 총 판매 유닛 수는?"

Relational schema에서 product table은 vendor_ID라는 외래 키를 가지게 된다. 그렇기에 이를 이용해먼저 각 vendor가 공급한 제품 수를 계산하고, 가장 많은 제품을 공급하는 vendor를 찾는다. 그 후, 해당 vendor가 제공한 제품들의 판매 유닛 수는 sales_transaction table과 join하여 total_amount를 더해 계산한다.

3-5) 각 매장에서 재주문 임계값 이하로 떨어져 재주문이 필요한 제품은?

relational schema에서 Inventory relation은 store_ID와 UPC를 foreign key이자 primary key로 가지게 된다. 또한, current_quantity와 reorder_threshold 속성을 가지기에, current_quantity가 reorder_threshold보다 작은 Inventory 항목을 찾으면 된다.

3-6) 로열티 프로그램 고객이 커피와 함께 주로 구매하는 상위 3개 품목은?

우선 sales_transaction table에서 loyalty_ID가 null이 아니여야 하며 Relational schema로 변환 시 transaction_ID와

UPC를 복합 PK(Primary Key)로 가지는 Transaction_Product라는 새로운 테이블이 생성될 것이다. 이것을 이용하여 커피를 포함하는 거래들을 식별하고, 해당 거래에서 커피가 아닌 다른 제품들을 계산한다.

3-7) 프랜차이즈 소유 매장 중 가장 다양한 제품을 제공하는 곳은?

Store 테이블과 Inventory 테이블을 조인하여 Store의 ownership_type으로 필터링한 뒤, Inventory 항목의 수를 계산하여 가장 다양한 제품을 제공하는 매장을 찾아낸다.