CSE4110 – Database System

Project1. E-R design and Relational Schema design

Spring 2021

학 과 : 컴퓨터공학과

학 번 : 20161634

이 름 : 장 수 길

목 차

1 프로젝트 개요

이번 프로젝트의 주제는 본인이 자동차 회사의 DB 매니저가 되었다고 가정하여 요구사항을 분석하고, 주어진 질의에 적절하게 대응할 수 있는 데이터베이스를 설계하기 위해 E-R model과 Relational Schema diagram을 이용하는 것이다.

2 E-R diagram

2.1 Entities

2.1.1 Company

Primary Key: company_ID

과제의 조건은 하나의 Company에 대한 DB를 만드는 것이었지만, 약간 확장하여 회사 자체를 하나의 Entity로 설계하였다. 회사의 이름은 항상 고유한 경우를 가정하여 primary key로 회사의 이름을 지정하였다.

2.1.2 Brand

Primary Key: Brand_ID

Brand의 경우, brand에 대해서 identification number를 primary key로 지정하였다. Brand의 ID로 어떤 brand인지 다시 식별하기 위해서 추가적인 attribute로 brand_name을 추가하였다.

2.1.3 Model

Primary Key: Model_ID

마찬가지로 각 모델에 대한 ID를 primary key로 지정하였고, 모델명을 식별하기 위해 model_ID라는 attribute를 추가하였다.

2.1.4 Plant

Primary_Key: plant_ID

자동차를 생산하는 플랜트를 하나의 entity로 설정하였다. 주어진 query 조건에 의하면 supplier 또한 plant들을 소유하는데, 설계 단계에서 회사소유의 plant와 supplier소유의 plant를 구별할 이유가 없다고 판단하여 단일 plant entity를 만들고, plant_owner이라는 attribute를 통해 plant의 소유자를 식별할 수 있도록 구성하였다.
Unique한 식별을 위해 마찬가지로 primary key는 plant_ID를 주었다.

2.1.5 Parts

Primary Key: Parts_ID

플랜트가 모델에 제공하는 부품을 특정하기 위해 primary_key로 parts_ID를 갖는다. 제조일자와 공급일자를 attribute로 추가하였다.

2.1.6 Options

Primary Key: option_ID

각 자동차에 붙는 옵션을 구별하기 위해서 primary key로는 option_ID를 설정하였다. 조건에 명시된 바와 같이 추가적인 attribute로는 색깔과 엔진 배기량, 그리고 옵션의 가격을 추가하였다.

2.1.7 Asset

Primary Key: VIN

자동차 하나의 개체를 구별해주는 entity를 설계하였다. primary key로는 VIN을, 추가적인 attribute로는 가격을 설정하였다.

2.1.8 Dealers

Primary Key: Dealer_ID

판매자 객체를 설계하였다. 동명이인이 있을 수 있다는 점을 고려하여 primary key는 이름이 아닌 ID로 설정하였고, dealer가 소유한 시점부터 차는 inventory에 들어간다고 가정하였다.

2.1.9 Customer

Primary Key: Customer_ID

손님 객체는 조건에 주어진 대로 전화번호, 이름, 주소, 성별, 그리고 연수입을 attribute로 추가시켰다. 전화번호는 multiattribute, 주소와 이름의 경우 complex attribute로 설정하였으며, primary key는 customer_ID로 설정하였다.

2.2 Relationships

2.2.1 Company – Brand [Owns_brand]

하나의 회사는 여러 개의 브랜드를 가질 수 있지만, 하나의 브랜드는 단 하나의 회사를 가져야만 하기 때문에 one-to-many total participation relationship으로 설계하였다.

2.2.2 Brand - Model [from_brand]

마찬가지로 하나의 브랜드는 여러 개의 모델을 가질 수 있고, 하나의 모델은 하나의 브랜드를 가져야만 하기 때문에 one to many total participation relationship으로 설계하였다.

2.2.3 Plant – Parts [manufactures]

하나의 플랜트는 여러종류의 부품을 생산할 수 있다고 가정하였다. 또한 부품은 여러곳의 플랜트에서 생산이 가능하다고 생각했다. 따라서 many to many relationship이며, 플랜트 없이 생산된 부품은 없기 때문에 total participation으로 설정하였다.

2.2.4 Parts – Models [supplies]

하나의 부품은 여러 개의 모델로 공급될 수 있는 부품이라고 가정하였으며, 하나의 모델은 여러 개의 부품으로 구성될 수 있다고 생각하였다. 따라서 many to many이며 모델이 없는 잉여 부품이 존재할 수 있으므로 partial participation을 설정하였다.

2.2.5 Model – Options [provide_options]

하나의 Option은 반드시 하나의 Model을 갖고, 하나의 Model은 여러개의 Option을 가질 수 있기 때문에 마찬가지로 one to many, 옵션은 반드시 모델이 존재해야 하기 때문에 total participation을 구성하였다.

2.2.6 Options – Asset [has_options]

하나의 옵션은 반드시 자동차의 asset을 가지지만, 자동차가 옵션을 가지지 않을 수 있기 때문에 Option → Asset의 경우 1...1, 반대의 경우 0..1의 cardinality로 표기하였다.

2.2.7 Asset – Dealers [stocks_vehicles]

하나의 자동차 asset은 하나의 dealer를 갖지만, 한명의 dealer는 여러개의 자동차 asset을 입고할 수 있기 때문에 one to many relationship으로 설정하였으며, dealer가 아무것도 입고하지 않는 경우를 생각해 partial participation으로 구성하였다. 입고일자를 표기하기 위해서 relation에 stock_date라는 attribute를 추가하였다.

2.2.8 Dealers – Customer [sells_vehicle]

손님은 여러명의 dealer에게 자동차를 구입할 수 있으며, dealer 또한 여러명의 손님에게 자동차를 판매할 수 있기 때문에 many to many relationship으로 표기하였으며, relationship에 판매 날짜와 자동차의 ID를 기록하기 위해 attribute로서 추가하였다.

3 Relational Schema

3.1 Entities with Relationships

효과적으로 각 entity와 그에 연결되는 entity들의 관계를 설명하기 위해 e-r diagram에서의 설명과 다른 방식으로 서술하겠다.
각 entity의 foreign key, primary key, cardinality, type, null 허용 여부와 entity 자체에 대해서 통합적으로 설명하도록 하겠다.

3.1.1 Company [Company-Brand]

회사의 경우 고유한 이름을 가지고 있기 때문에 primary key를 이름으로 설정하였고, cardinality 또한 e-r diagram에서의 설명과 동일한 이유로 one-to-many이며, 이름이라는 primary key하나만으로 독립적으로 식별이 가능하기 때문에 brand와의 관계는 non-identifying relationship이며, 회사가 없는 brand는 존재하지 않기 때문에 null값을 허용하지 않았다.

3.1.2 Brand [Brand-Model]

브랜드는 브랜드 ID를 통해 식별하였으며, 모델과 관계를 가지고 있다. 마찬가지로 독립적으로 brand_ID를 통해 식별이 가능하므로 non-identifying relationship이며, 브랜드가 없는 모델이 존재할 리 없기 때문에 null값을 허용하지 않았고, one to many로 설정하였다.

3.1.3 Model [Model-Options]

모델도 Model_ID를 통해 식별하였다. 모델은 옵션과 관련이 있는데, Model_ID만으로 고유한 식별이 가능한 모델에 여러가지 옵션이 존재할 수 있고, 옵션은 반드시 모델이 존재해야 하므로 null값을 허용하지 않는 non identifying one-to-many relation으로 설정하였다.

3.1.4 Plant [Plant-Manufactured]

Plant는 부품을 생산해주는 집합이며, 하나의 plant에서 여러가지 부품을 만들 수 있다. 즉 하나의 plant에서는 다양한 부품을 생산하고 하나의 생산이 여러가지의 plant를 가질 수는 없기 때문에 생산(manufactured) entity와는 one-to-many relationship을 설정하였고, plant없이 생산은 불가능하기 때문에 null값을 허용하지 않았다. Manufactured entity는 Plant ID없이는 고유한 식별이 불가능하기 때문에 identifying relationship이다.

3.1.5 Manufactured [Manufactured – Parts]

생산항목들을 저장하는 entity이며, plant와 parts와 관계가 존재한다. plant-manufactured의 관계와 동일하게 manufactured-parts 관계는 하나의 생산 항목으로 하나의 부품만을 지원하기 때문에 one-to-many relationship, parts없는 생산은 이루어지지 않으므로 null값을 허용하지 않았으며, Parts_ID없이는 고유한 식별이 불가능하므로 identifying relationship이다. Primary Key는 둘 다 외래키인 parts_ID와 plant_ID를 사용한다.

3.1.6 Parts [Parts -Supplied]

부품들을 저장하는 entity이며, manufactured와 동일하게 supplied와 관계를 이루고 있다. 하나의 부품으로 여러종류의 모델에 공급이 가능하지만, 한번의 공급이 여러 개의 부품으로 이루어질 수는 없기때문에 one-to-many relationship 을 사용하였다. parts_id 없이 식별이 불가능하므로 identifying relationship을 구성하였다.

3.1.7 Supplied [Supplied-Model]

"공급"항목을 저장하는 entity이며, 하나의 parts로는 여러 개의 공급이 가능하지만 하나의 공급은 하나의 부품으로만 이루어지기 때문에 one-to-many relationship을 사용하였고, manufactured entity와 동일한 이유로 identifying relationship으로 설정하였다. Primary Key 또한 동일하게 parts_ID와 model_ID이다.

3.1.8 Options [Options-Asset]

특정 모델에서 지원하는 옵션들을 저장한다. 따라서 Primary key로는 외래키인 model_ID가 포함되며, Option_ID 또한 사용한다.

Asset entity와 관계를 가지고 있는데, 하나의 옵션은 여러 개의 asset에 공급이 가능하지만 하나의 asset은 하나의 option만 보유 가능하므로 one-to-many relationship 이며, 하나의 asset은 option이 존재하지 않아도 가능할뿐더러, vehicle_ID를 통해 고유한 식별이 가능하므로 non-identifying, allowing null relationship이다.

3.1.9 Asset [Asset – Dealers]

자동차 개체를 저장한다 따라서 자동차 번호로 primary key가 구성되며, 가격과 입고날짜, 모델, 브랜드명 등을 속성으로 가진다. 입고한 딜러와 관계를 가지고 있으며 서로 독립적인 식별이 가능하기 때문에 non-identifying이지만, 딜러 없이 입고는 불가능하므로 null을 허용하지 않았다. 한명의 딜러는 여러 개의 asset을 보유 가능하므로 one-to-many relationship을 채택하였다.

3.1.10 Dealers [Dealers – Sales]

딜러를 저장한다. 딜러의 이름은 동명이인이 가능하므로 딜러 ID를 개별적으로 만들어서 식별한다. Sales Entity와 관계를 갖는데, 한명의 딜러가 여러명의 sales를 이룰 수는 있지만 하나의 sales가 여러명의 dealer는 허용하지 않으며, dealer없는 sales는 존재하지 않기 때문에 one-to-many, null-disallowing relationship이다. Sales는 독립적으로 vehicle_ID를 통해 식별하므로 non-identifying relationship 이다.

3.1.11 Sales [Sales-Customer]

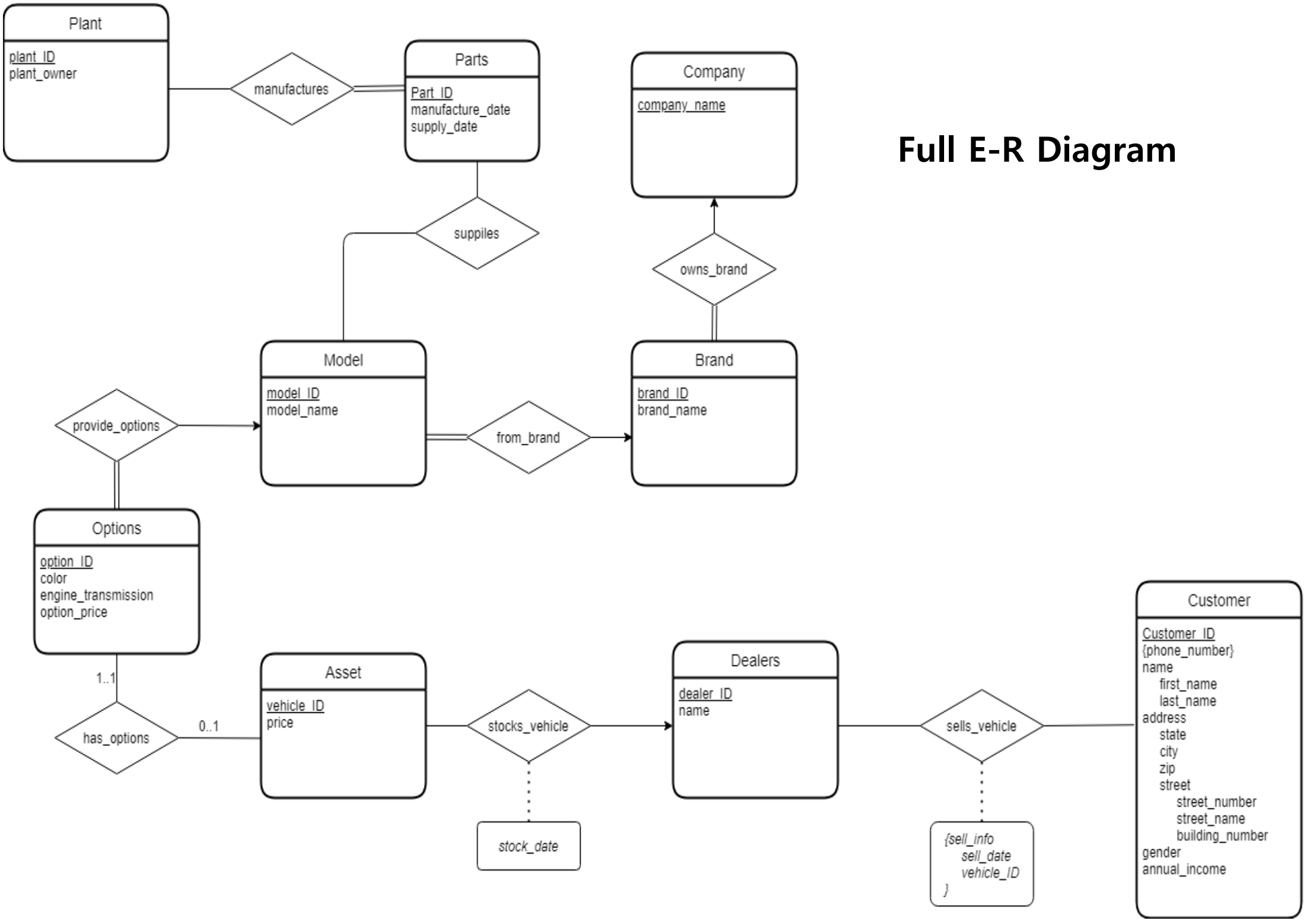
판매 하나하나를 저장한다. primary key로는 vehicle_ID만을 갖는다. customer와 관계를 갖는데, customer는 customer_ID를 통해 식별이 가능하므로 non-identifying이며, customer가 존재하지 않을 수 없으므로 null-disallowing, 그리고 한명의 customer는 여러 개의 sales에 포함되는 것이 가능하므로 one-to-many relationship 이다.

3.1.12 Customer_ID [Customer-Customer_Phones]

고객 정보를 저장한다. 고유한 식별을 위해 고객 ID를 사용하고, 고객에 대한 다양한 정보 (성별, 연봉, 주소) 등을 저장한다. Customer_Phones entity와 연결되어 있는데, 이유는 한명의 고객은 여러 개의 전화번호를 가지는 것이 가능하기 때문이다. 따라서 one-to-many relationship 이며, 전화번호는 단독으로 고객을 식별하는것이 불가능하기때문에 identifying relationship으로 설정하였다.

3.1.13 Customer_phones

고객의 전화번호를 저장한다. 앞서 언급한 바와 같이 전화번호는 단독으로 고객을 식별하는 것이 불가능하기 때문에 primary key로 customer_ID가 추가되었다.



Full E-R Diagram

Full Relational Schema Diagram