**데이터베이스설계**

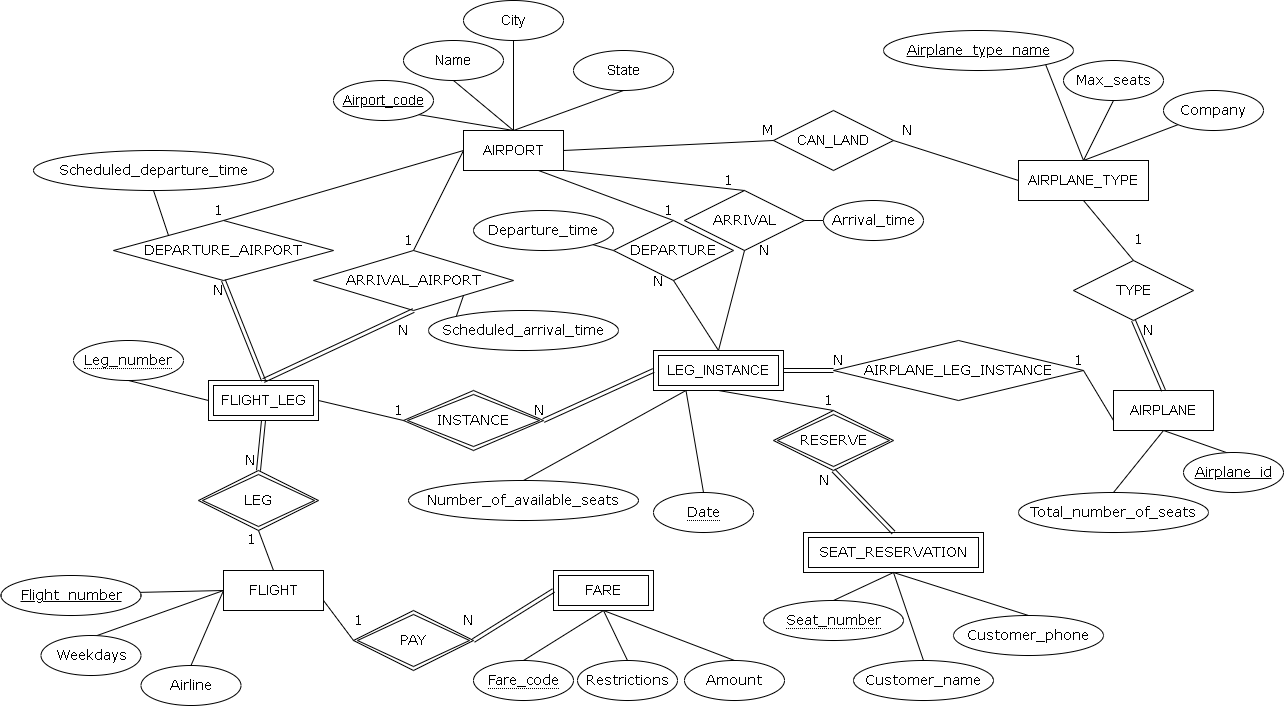
설계 프로젝트: AIRLINE DATABASE

12172065 길예림

1. 프로젝트 개요

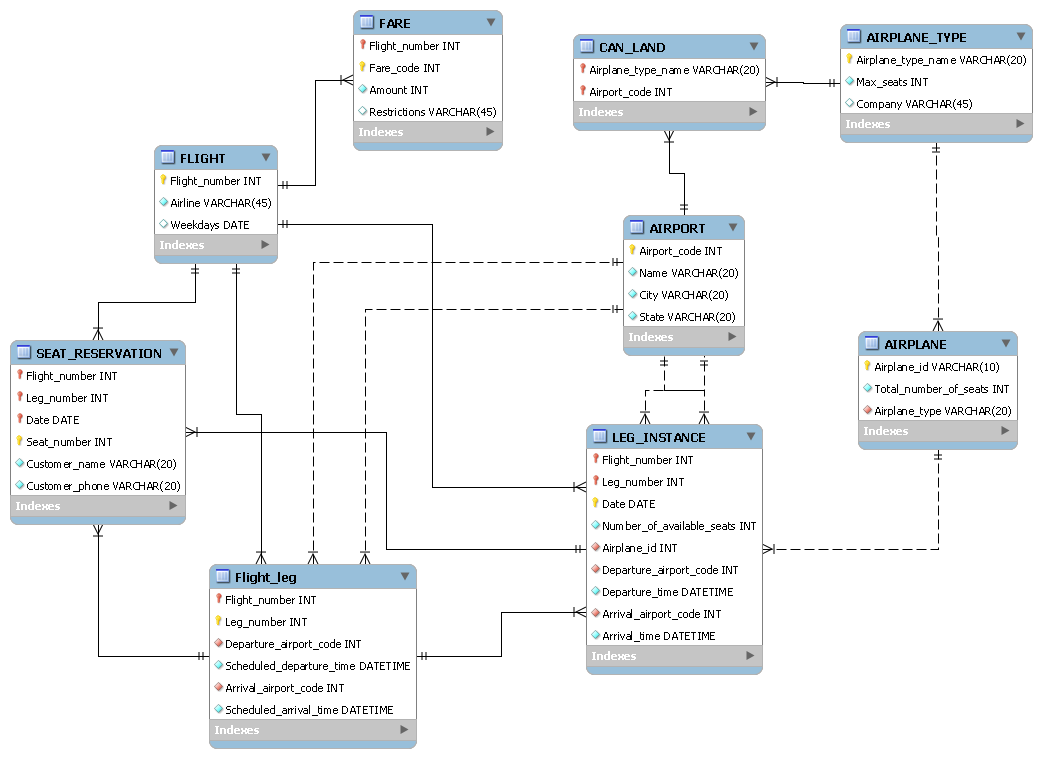
해당 프로젝트는 주어진 schema를 바탕으로 AIRLINE 관련 데이터베이스를 설계 및 관리 페이지를 설계하는 것입니다. 주어진 테이블 중 Entity와 Relationship이 될 것을 구분하고 각 테이블마다의 관계 및 구조를 E-R Diagram으로 표기합니다. 이를 바탕으로 MySQL에 구현하고 workbench에서 EER Diagram을 설계한 다음, 관리 페이지를 Java, HTML로 구현합니다. 관리 페이지는 정보를 입력, 수정, 삭제할 수 있는 관리자 페이지와 예약 관련 일을 행할 수 있는 사용자 페이지로 구분하여 설계합니다.

1. E-R Diagram



객체끼리와의 관계를 바탕으로 E-R Diagram을 직접 작성하였습니다. 이를 바탕으로 MySQL workbench에서 EER Diagram을 작성하였습니다.

1. MySQL
   1. Workbench – EER Diagram



제공된 schema를 바탕으로 MySQL workbench에서 작성한 EER Diagram입니다. FLIGHT 개체를 기준으로 시계 방향으로 설명드리겠습니다.

FLIGHT 개체의 PK는 Flight\_number (비행 번호)이며 Airline (항공사)과 Weekdays (휴일)는 기본 속성입니다. FK는 존재하지 않습니다. 제약조건에 의해 FLIGHT\_LEG와는 1:N 관계입니다. 비행 행위에 있어 항공사는 필수로 생각되어 Airline은 not null로 지정해주었으며, Weekdays (비행을 하지 않는 날짜, 휴일)는 없을 수도 있다고 가정하여 null 값을 허용하였습니다.

FARE 개체의 PK는 Flight\_number, Fare\_code (요금 코드)이며 이 중 Flight\_number는 FLIGHT 개체의 PK로부터 가져온 FK입니다. 따라서 식별관계가 됩니다. 한 항공기에도 여러 가지 조건에 의해 좌석의 값은 달라지므로 여러 개의 요금이 존재할 수 있다고 가정하여 1:N 관계로 설정해주었습니다. 요금에서 Amount (값)는 필수라고 생각되어 not null로 지정해주었으며 Restrictions (제약조건)은 없을 수도 있다고 가정하여 null을 허용하였습니다.

CAN\_LAND는 AIRPLANE\_TYPE 개체와 AIRPORT 개체를 이어주는 관계로 가정하였습니다. 따라서 보유한 2개의 속성은 각각 AIRPLANE\_TYPE, AIRPORT의 PK를 FK로 삼았습니다. 또한 연결되는 두 개체는 M:N의 관계로 가정하였으므로 위와 같이 표기하였습니다.

AIRPLANE\_TYPE 개체의 PK는 Airplane\_type\_name (항공기 종류 이름) 이며 Max\_seats (최대 좌석수), Company (항공기를 만든 회사)는 기본 속성입니다. Max\_seats를 알아야 예약을 받을 수 있으므로 해당 속성은 not null로 지정해주었고 항공기를 만든 회사는 모를 수도 있으며, 모르더라도 항공 자체에 큰 영향을 끼치지 않는다고 생각되어 null을 허용해주었습니다. AIRPLANE 개체와는 1:N의 관계로 지정해주었습니다. 왜냐하면 AIRPLANE (항공기)은 여러 종류 중 하나로 무조건 정해지지만, 항공기 종류의 입장에서 보았을 때는 해당 종류의 항공기가 여러 대 있을 수 있기 때문입니다.

AIRPORT 개체의 PK는 Airport\_code (공항 코드)이며 Name (공항 이름), City (공항이 위치한 도시), State (공항이 위치한 주)는 기본 속성입니다. 공항의 전체적인 위치는 필수적이므로 모두 not null로 지정해주었습니다. LEG\_INSTANCE 개체와의 관계는 1:N으로 설정하였습니다. 왜냐하면 LEG\_INSTANCE의 속성 중 Departure\_airport\_code (출발하는 공항의 코드)와 Arrival\_airport\_code (도착하는 공항의 코드)는 항공편에 따라 달라지기 때문입니다. 또한 FLIGHT\_LEG 개체와도 동일하게 1:N의 관계로 설정해주었습니다. 왜냐하면 FLIGHT\_LEG (비행 구간)의 수에 따라 도착하는 공항, 출발하는 공항 등 여러 공항이 필요하게 참여하게 되기 때문입니다.

LEG\_INSTANCE 개체의 PK는 Flight\_number, Leg\_number (비행 구간 번호), Date (비행 날짜)이며 이 중 Flight\_number는 FLIGHT 개체의 PK로부터 가져온 FK, FLIGHT\_LEG 개체의 PK로부터 가져온 FK입니다. Number\_of\_available\_seats (사용가능한 좌석 수)를 알아야 비행 예약이 가능하므로 필수적이라고 생각되어 not null로 지정해주었습니다. Airplane\_id (항공기 id)는 AIRPLANE 개체의 PK로부터 가져온 FK입니다. 항공기를 알아야 예약이 가능하므로 not null로 지정해주었습니다. Departure\_airport\_code와 Arrival\_airport\_code 도 동일한 이유로 not null로 지정해주었으며 둘 다 AIRPORT 개체의 PK인 Airport\_code를 FK로 지정한 것입니다. Departure\_time (비행 출발 시간)과 Arrival\_time (비행 도착 시간)은 필수적이라 생각되어 not null로 지정해주었습니다. SEAT\_RESERVATIOH 개체와는 1:N의 관계로 지정해주었습니다. 왜냐하면 한 항공편에 좌석 예약은 여러 개 존재하는 것이 당연하기 때문입니다.

FLIGHT\_LEG 개체의 PK는 Flight\_number, Leg\_number이며 이 중 Flight\_number는 FLIGHT 개체의 PK로부터 가져온 FK입니다. 기본 속성 중 Departure\_airport\_code와 Arrival\_airport\_code는 AIRPORT 개체의 PK인 Airport\_code의 FK입니다. 모든 속성이 필수라 생각되어 not null로 지정해주었으며, SEAT\_RESERVATION 개체와 LEG\_INSTANCE 개체와는 각각 1:N 관계로 설정하였습니다. 왜냐하면 한 비행구간을 지나는 항공편의 경우 좌석 예약이 여러 개인 것은 당연하며, LEG\_INSTANCE는 말 그대로 FLIGHT\_LEG를 이루는 구성요소이기 때문입니다.

마지막으로 SEAT\_RESERVATION 개체의 PK는 Flight\_number, Leg\_number, Date, Seat\_number이며 이 중 Flight\_number는 FLIGHT 개체의 PK를 FK로, Leg\_number는 FLIGHT\_LEG 개체의 PK를 FK로, Date는 LEG\_INSTANCE 개체의 PK를 FK로 지정한 것입니다. 예약이 잡힌다면 예약한 고객의 이름과 핸드폰 번호는 알아야 한다고 생각되어 not null로 지정해주었습니다.

* 1. 생성 스크립트 (forward engineering)

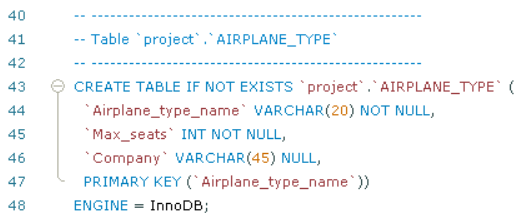
MySQL workbench에서 작성한 EER Diagram을 forward engineering 했을 때 얻을 수 있는 생성 스크립트입니다. 전체 생성 스크립트는 길어서 프로젝트 폴더의 ‘생성스크립트.docx’에 텍스트로 저장하였습니다. 보고서에는 이 중 일부인 create table, create index, create view문만 캡쳐본으로 첨부하였습니다.

* + 1. CREATE TABLE

테이블이(가) 표시된 사진

자동 생성된 설명텍스트이(가) 표시된 사진

자동 생성된 설명

`

텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명

테이블이(가) 표시된 사진

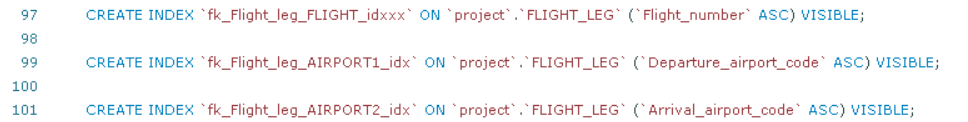
자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명

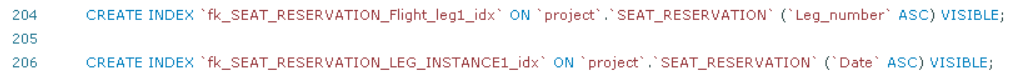
* + 1. CREATE INDEX











* + 1. CREATE VIEW

텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명



1. JavaScript & HTML
   1. 기본 설정

처음 접속했을 때, 관리자와 사용자를 구분하기 위해 처음 접속했을 시, id와 password 입력을 하도록 설정하였습니다.

테이블이(가) 표시된 사진

자동 생성된 설명

MySQL의 USER 테이블을 따로 선언해주었으며 해당 테이블의 아이디와 비밀번호와 동일할 때 관리자 페이지 또는 사용자 페이지에 접근 가능합니다. 존재하지 않는 아이디, 비밀번호를 입력하면 로그인 실패창이 뜹니다.

텍스트이(가) 표시된 사진

자동 생성된 설명텍스트이(가) 표시된 사진

자동 생성된 설명

manager 또는 user로 접속했을 때 원하는 화면을 얻지 못하였습니다.

1. 실행화면
   1. MySQL

MySQL workbench에서 forward engineering 이후, 제대로 생성되었는지 확인하기 위해 MySQL에서 직접 결과를 확인하였습니다.

텍스트이(가) 표시된 사진

자동 생성된 설명 텍스트이(가) 표시된 사진

자동 생성된 설명

EER Diagram을 통해 만든 database와 table들이 모두 입력되어 있는 것을 확인할 수 있습니다.

텍스트이(가) 표시된 사진

자동 생성된 설명텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

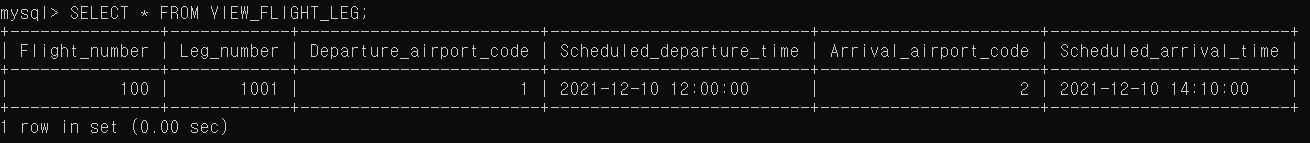
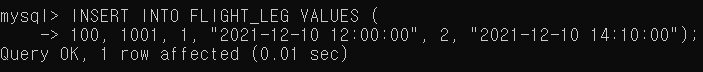
자동 생성된 설명텍스트이(가) 표시된 사진

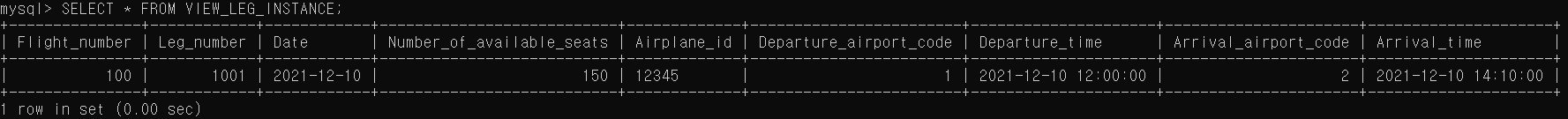
자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명텍스트이(가) 표시된 사진

자동 생성된 설명





텍스트이(가) 표시된 사진

자동 생성된 설명텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명텍스트이(가) 표시된 사진

자동 생성된 설명

모두 제대로 값이 들어가는 것을 확인할 수 있었습니다.

1. 고찰

한 학기 동안 데이터베이스 및 자바 스크립트, html을 배우면서 설계 프로젝트를 수행할 수 있을 정도의 내용은 모두 배웠으나 아직 제가 혼자 프로젝트를 구상하고, 홈페이지를 구상하고 설계하기에는 부족하다는 생각이 들었습니다. 이해한 바를 바탕으로 나름의 관리자 페이지와 사용자 페이지를 작성하였지만 실제로 구성하는 것에는 실패하였고 정확한 원인을 발견하지 못해 해결하지 못했습니다.

다만, MySQL workbench를 통해 주어진 schema를 바탕으로 테이블을 설계하는 것은 어느 정도 성공적이라고 생각합니다. 주어진 제약 조건과 요구 사항을 모두 만족했다고 생각하고, 이를 만족시키기 위해 어떻게 EER Diagram을 작성해야 할지, 어떤 순서로 값을 insert 해야 할지, transaction을 어떻게 구현해야 할지 등을 혼자 고민해보고 설계함으로써 DB에 대한 이해도가 더 상승하였고 사고력도 증진되었다고 느꼈습니다.

또한, sql.js, home.js, manager.js는 어느 정도 구현에 성공하였다고 생각합니다. 특히 home의 경우, 아이디 비밀번호를 입력하였을 때 관계자 페이지 또는 사용자 페이지로 넘어가는 것은 생각했던 대로 구현하였습니다. 아쉬운 것은 관리자 페이지 (/manager) 구현에 실패하여 조회할 수 없다는 것입니다. Cannot GET /manager 멘트가 뜨는 것과 F12로 개발자 도구를 확인해보았을 때 404 (Not Found)가 뜨는 것으로 보아 단순히 오타가 있어 구현이 안 될 것일 수도 있고, /를 통해 페이지가 넘어가는 것을 제대로 구현하지 못했을 수도 있을 것입니다. 개인적으로 가장 큰 이유라고 생각되는 것은 /manager (manager.js)에 너무 많은 기능을 한꺼번에 구현하려고 해서 충돌이 일어났을 것이라는 점입니다. manager로 로그인한 후에 관리자 페이지에서 어떤 테이블을 삽입/수정/삭제할 것인지 택하고 그 다음 그 행위를 수행하는 페이지를 또 따로 만들었다면 구현 가능했을 것이라고 추측합니다. 또한, 한 페이지에 Airport/Airplane/Flight 테이블을 삽입/수정/삭제하는 것을 구현하게 되었을 때 여러 개의 버튼이 있게 되는데 이를 어떻게 처리할 지에 대한 고민을 깊게 해보지 못한 것도 실패의 원인 중 하나라고 생각합니다.