#데이터베이스 설계 프로젝트

12142058

이재용

1. 프로젝트 개요

1)목표 : 항공기 예약 시스템 설계

2)요구 기능 :

- 아래 두개의 페이지 구성

1. 항공기 및 비행 정보 관리하는 관리자 페이지

2. 예약조회 및 예약, 예약확인을 위한 사용자 페이지

1. 주어진 요구사항 외 두 기능을 위한 요구사항 추가 후 ERD 완성

2. 관리자 페이지

1. 공항 정보 입력/수정/삭제

2. 항공기 정보 입력/수정/삭제

3. 항공편 정보 입력/수정/삭제

4. 위 기능을 구현하기 위한 제약조건을 제시 (정규화 확인)

3. 사용자 페이지

1. 예약하기, 예약조회, 예약취소 기능

2. 위 기능을 구현하기 위한 제약조건, 색인, 트랜잭션 등을 제시

기본적인 요구사항

* 각각의 FLIGHT는 Flight\_number, Leg\_numbers(1, 2, 3)와 함께 하나 이상의 FLIGHT\_LEGs을 포함
* 각각의 FLIGHT\_LEG는 arrival times, departure times, airports 그리고 하나 이상의 LEG\_INSTANCE을 포함
* 요금은 각각의 항공권에 포함
* 각각의 FLIGHT\_LEG instance, 에는 leg에 사용된 항공기와 실제 도착, 출발시간 및 공항이 을 포함
* AIRPLANE은 Airplane\_id에 의해 식별되며 특정 AIRPLANE\_TYPE에 속함
* CAN\_LAND는 착륙할 수 있는 공항과 AIRPLANE\_TYPE에 연관되어 있음
* AIRPORT는 Airport\_code에 의해 식별된다.

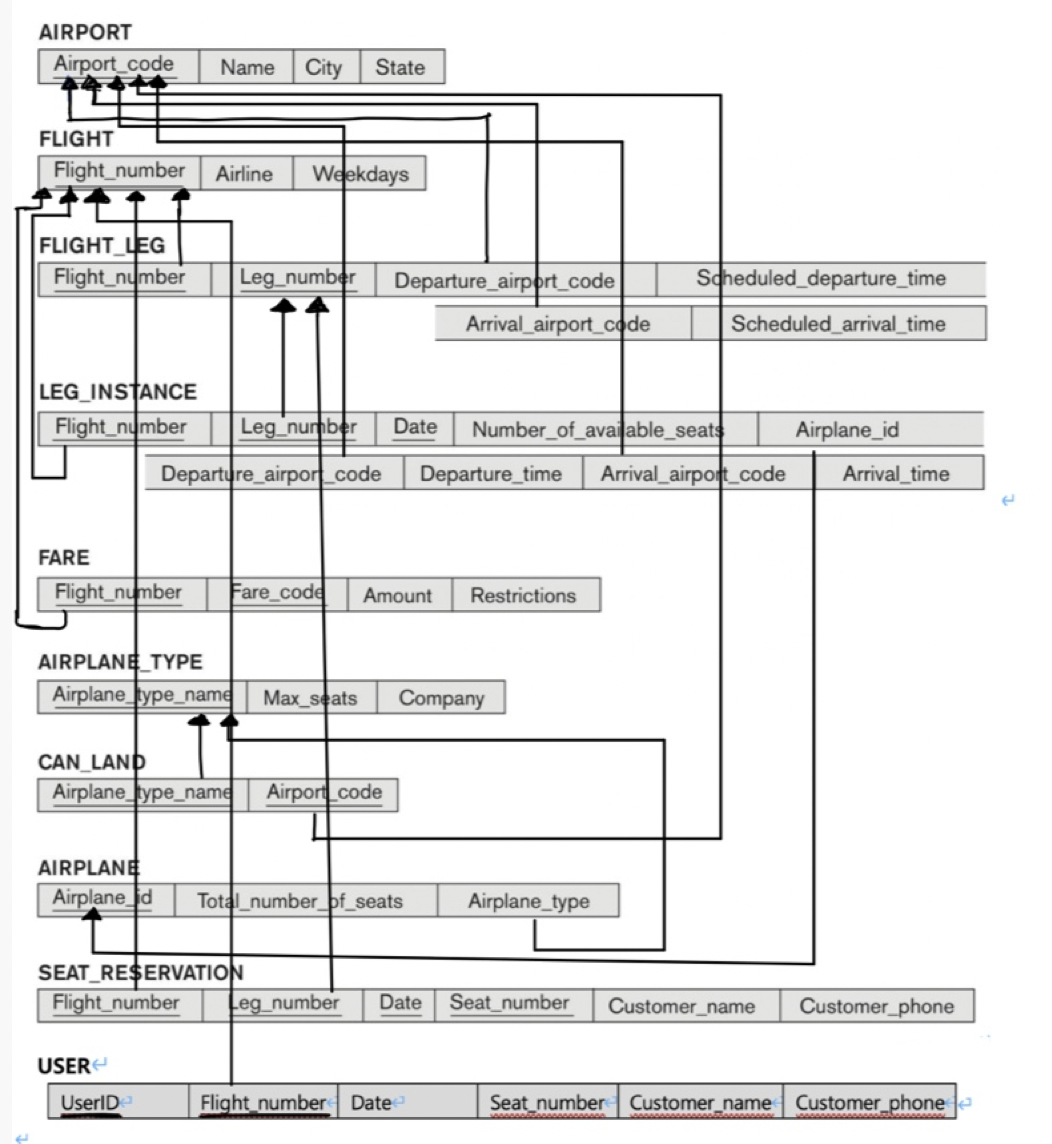
기본적인 제약조건

* FLIGHT\_LEG 테이블에는 flight number, leg number, arrival airport code 그리고 departure가 존재. 만약 존재하지 않을 경우 FLIGHT\_LEG 테이블을 먼저 삽입
* AIRPLANE 관계에는 airport id가 존재
* Airplane\_id는 airplane type을 제공한다. Airplane type에는 착륙할 공항 CAN\_LAND 관계 안에 튜플<Airplane\_type\_name, Airport\_code>이 존재하여야 함
* 사용 가능한 좌석 수는 AIRPLANE\_TYPE 관계의 좌석 수와 합해져야 함
* Flight number가 삽입된 FLIGHT 관계가 존재
* Departure airport code가 있는 AIRPORT 관계 존재
* Arrival airport code가 있는 AIRPORT 관계 존재

2. 상세 설계내용

- ER MAPPING

데이터베이스 생성 및 데이터 삽입을 위해 다음은 관계형 데이터베이스 스키마 를 ER매핑하여 다이어그램을 나타내었습니다. 추가적으로 USER Table을 만들어 관계형 데이터베이스 스키마 다이어그램을 나타내었습니다.



3)FLIGHT\_LEG에서 PK인 Flight\_number는 외래키로 FLIGHT테이블의Flight\_number를 받았고Departure\_airport\_code와 Arrival\_airport\_code는 AIRPORT 테이블의 Airport\_code를 받았습니다. 4)LEG\_INSTANCE테이블에서 Flight\_number는 외래키로FLIGHT의 Flight\_number를 받았고 FLIGHT\_LEG테이블에서Leg\_number또한 받았습니다. Departure\_airport\_code 및 Arrival\_airport\_code는 AIRPORT의 Airport\_code를 받았습니다.

5)FARE 테이블의 Flight\_number는 외래키로FLIGHT의 Flight\_number을 받았습니다.

7)CAN\_LAND테이블의 Airplane\_code는 AIRPORT테이블의 Airport\_code를 외래키로 설정하였으며 8)AIRPLANE 테이블의Airplane\_type은 AIRPLANE\_TYPE의 Airplane\_type\_name를 외래키로 설정하였습니다.

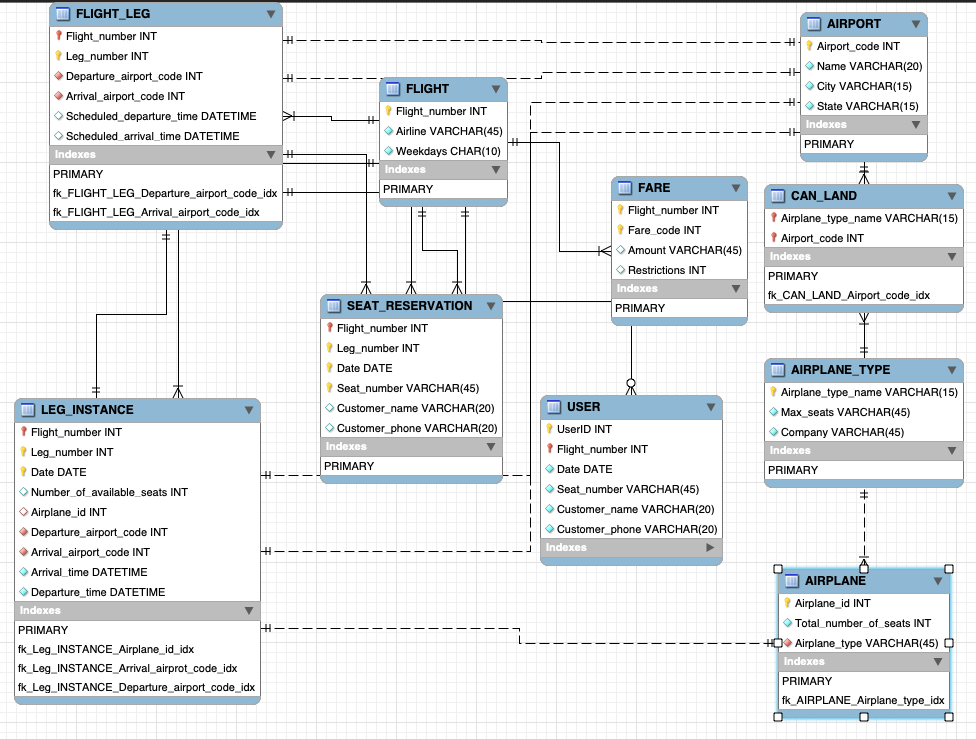
9)SEAT\_RESERVATION테이블에서 Flight\_number은 FLIGHT의 Flight\_number을 외래키로 설정하였으며 Leg\_number는 LEG\_NUMBER의 Leg\_number을 외래키로 설정하였습니다.

마지막으로 추가한 10)USER테이블에서 Flight\_number는 FLIGHT테이블의 Flight\_number를 외래키로 설정하였습니다.

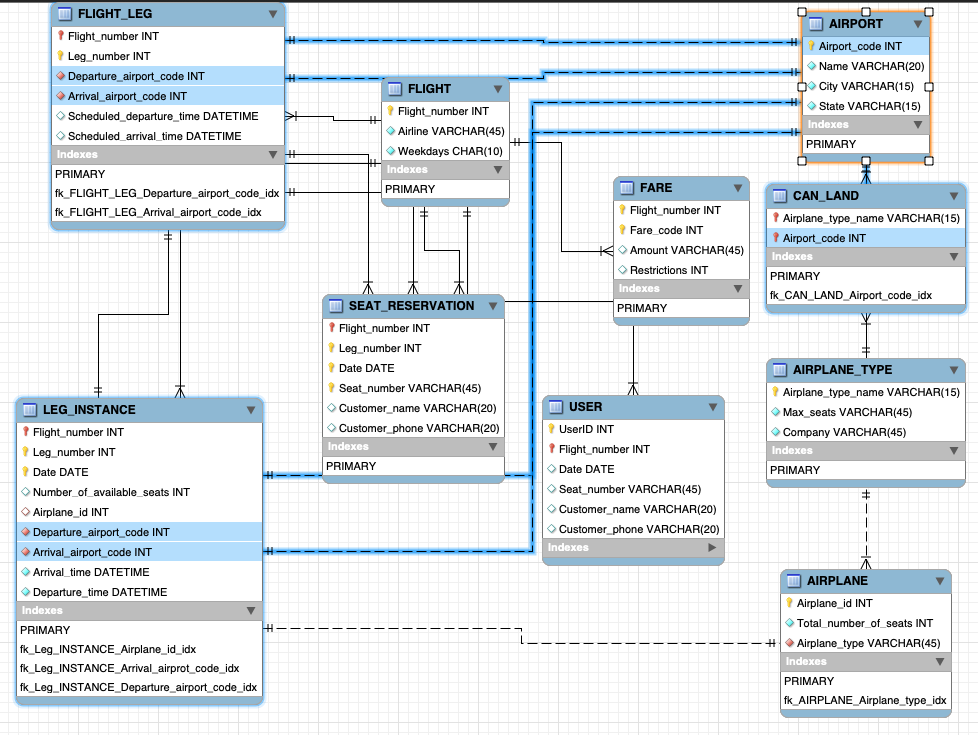
- EERDiagram

1) 전체 EER Diagram

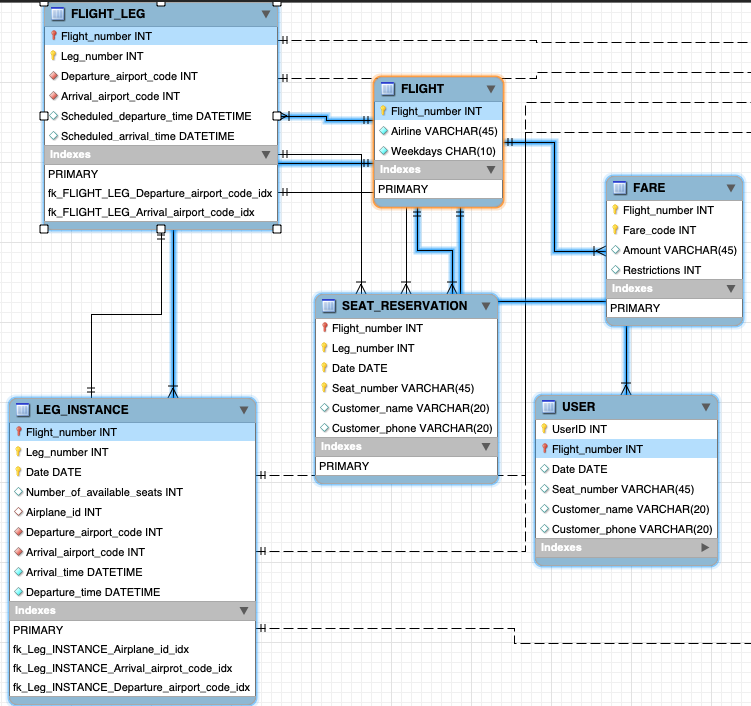
다음은 MySQL Workbench에서 작성한 EER Diagram 입니다.



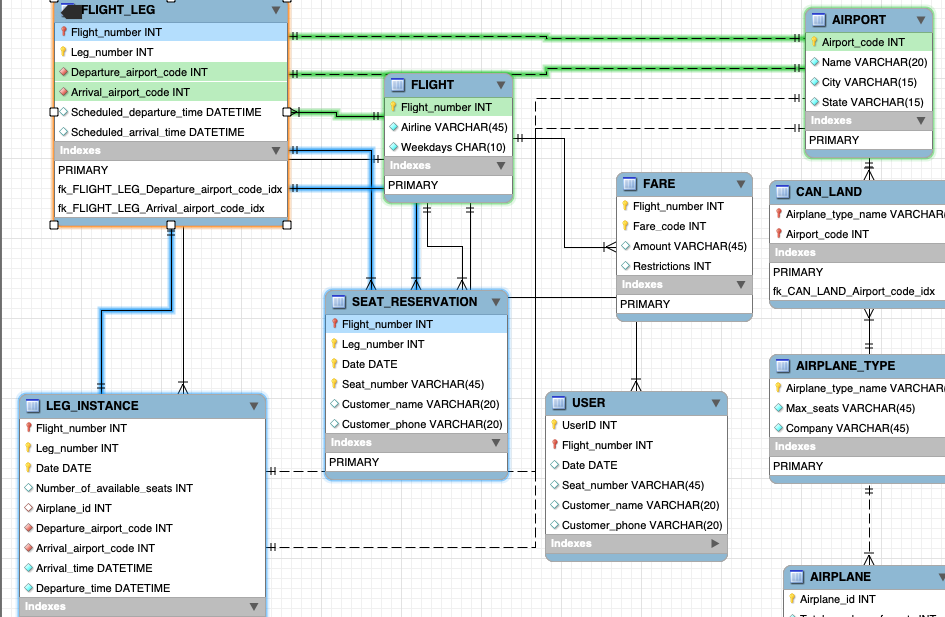
2) 세부 EER Diagram



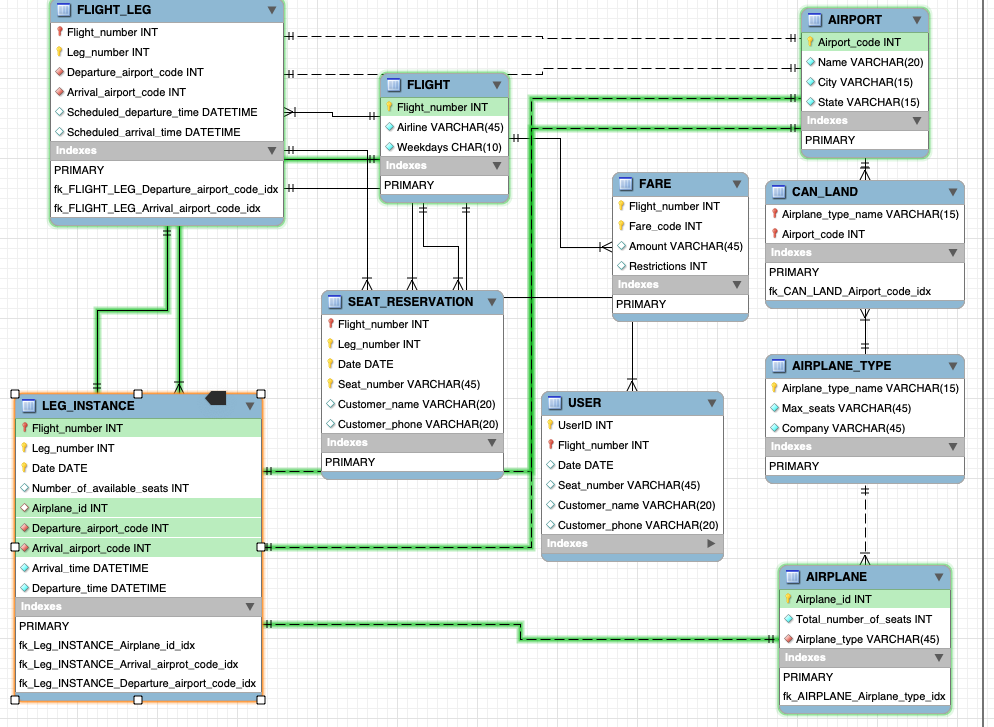
AIRPORT의 테이블은 FLIGHT\_LEG, LEG\_INSTANCE 테이블와는 비식별관계로 1:1로 연결해주었습니다. 그리고 CAN\_LAND테이블와는 1:N 식별 관계로 생성하였습니다.



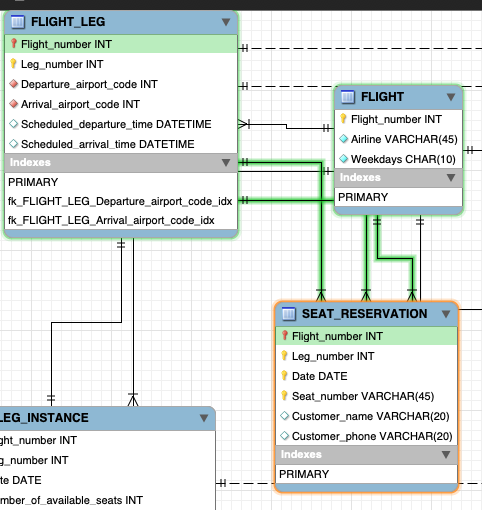
FLIGHT 테이블은 FARE, FLIGHT\_LEG, LEG\_INSTANCE, USER, SEAT\_RESERVATION으로 연결되어 있으며 전부 1:N 식별 관계로 연결해주었습니다.



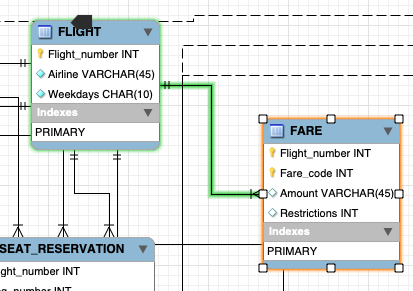
FLIGHT\_LEG테이블은 LEG\_INSTANCE와 Leg\_number로 연결되어 있으며 1:1로 식별 관계, SEAT\_RESERVATION테이블과 1:N 식별 관계로 지정해주었습니다.



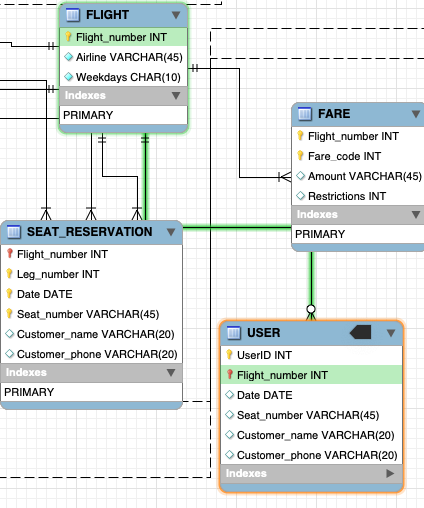
LEG\_INSTANCE 테이블은 FLIGHT\_LEG테이블과 1:1 식별 관계이며 AIRPORT, AIRPLANE테이블와 1:1 비식별 관계를 설정하였습니다.



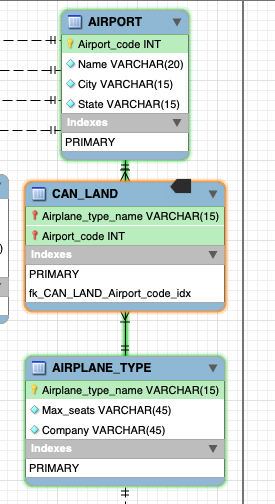
SEAT\_RESERVATION 테이블은 FLIGHT\_LEG테이블과 N:1 식별 관계로 설정하였습니다.



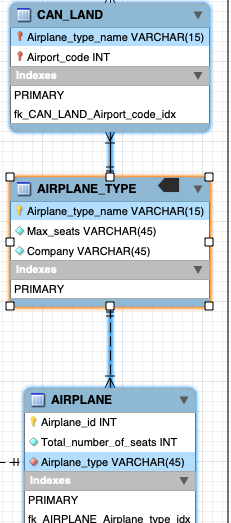
FARE테이블은 FLIGHT테이블과 N:1 식별 관계로 설정하였습니다.



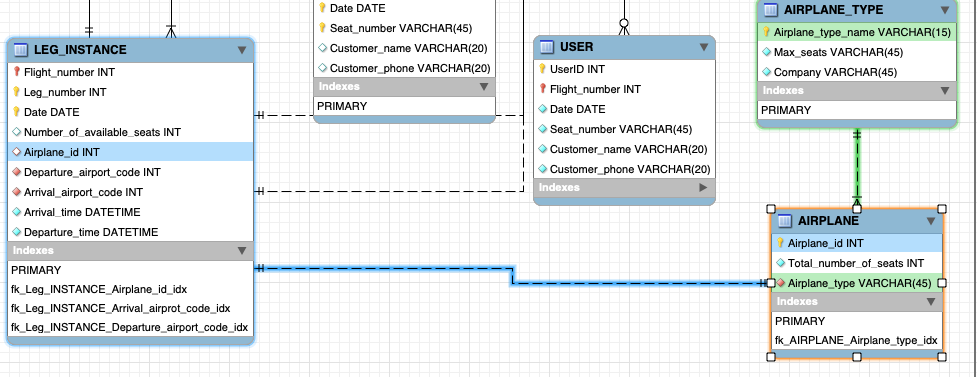
USER 테이블은 FLIGHT와 N:1 식별 관계이며 USER가 등록을 안할 경우를 생각하여 0을 포함하였습니다.



CAN\_LAND 테이블은 AIRPORT테이블, AIRPLANE\_TYPE테이블과 N:1 식별 관계로 설정하였습니다.



AIRPLANE\_TYPE 테이블은 CAN\_LAND테이블 1:N 식별 관계, AIRPLANE테이블과 1:N 비식별 관계로 설정하였습니다.



AIRPLANE 테이블은 AIRPLANE\_TYPE과 N:1 비식별 관계, LEG\_INSTANCE와 1:1 비식별 관계로 설정하였습니다.

3 AIRLINE 데이터베이스 생성 스크립트

CREATE SCHEMA IF NOT EXISTS `DATABASE\_PROJECT` DEFAULT CHARACTER SET utf8 ;

USE `DATABASE\_PROJECT` ;

-- -----------------------------------------------------

-- Table `DATABASE\_PROJECT`.`AIRPORT`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `DATABASE\_PROJECT`.`AIRPORT` (

`Airport\_code` INT NOT NULL,

`Name` VARCHAR(20) NOT NULL,

`City` VARCHAR(15) NOT NULL,

`State` VARCHAR(15) NOT NULL,

PRIMARY KEY (`Airport\_code`))

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `DATABASE\_PROJECT`.`FLIGHT`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `DATABASE\_PROJECT`.`FLIGHT` (

`Flight\_number` INT NOT NULL,

`Airline` VARCHAR(45) NOT NULL,

`Weekdays` CHAR(10) NOT NULL,

PRIMARY KEY (`Flight\_number`))

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `DATABASE\_PROJECT`.`FLIGHT\_LEG`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `DATABASE\_PROJECT`.`FLIGHT\_LEG` (

`Flight\_number` INT NOT NULL,

`Leg\_number` INT NOT NULL,

`Departure\_airport\_code` INT NOT NULL,

`Arrival\_airport\_code` INT NOT NULL,

`Scheduled\_departure\_time` DATETIME NULL,

`Scheduled\_arrival\_time` DATETIME NULL,

PRIMARY KEY (`Flight\_number`, `Leg\_number`),

INDEX `fk\_FLIGHT\_LEG\_Departure\_airport\_code\_idx` (`Departure\_airport\_code` ASC) VISIBLE,

INDEX `fk\_FLIGHT\_LEG\_Arrival\_airport\_code\_idx` (`Arrival\_airport\_code` ASC) VISIBLE,

CONSTRAINT `fk\_FLIGHT\_LEG\_Departure\_airport\_code`

FOREIGN KEY (`Departure\_airport\_code`)

REFERENCES `DATABASE\_PROJECT`.`AIRPORT` (`Airport\_code`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_FLIGHT\_LEG\_Arrival\_airport\_code`

FOREIGN KEY (`Arrival\_airport\_code`)

REFERENCES `DATABASE\_PROJECT`.`AIRPORT` (`Airport\_code`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_FLIGHT\_LEG\_Flight\_number`

FOREIGN KEY (`Flight\_number`)

REFERENCES `DATABASE\_PROJECT`.`FLIGHT` (`Flight\_number`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `DATABASE\_PROJECT`.`AIRPLANE\_TYPE`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `DATABASE\_PROJECT`.`AIRPLANE\_TYPE` (

`Airplane\_type\_name` VARCHAR(15) NOT NULL,

`Max\_seats` VARCHAR(45) NOT NULL,

`Company` VARCHAR(45) NOT NULL,

PRIMARY KEY (`Airplane\_type\_name`))

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `DATABASE\_PROJECT`.`AIRPLANE`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `DATABASE\_PROJECT`.`AIRPLANE` (

`Airplane\_id` INT NOT NULL,

`Total\_number\_of\_seats` INT NOT NULL,

`Airplane\_type` VARCHAR(45) NOT NULL,

PRIMARY KEY (`Airplane\_id`),

INDEX `fk\_AIRPLANE\_Airplane\_type\_idx` (`Airplane\_type` ASC) VISIBLE,

CONSTRAINT `fk\_AIRPLANE\_Airplane\_type`

FOREIGN KEY (`Airplane\_type`)

REFERENCES `DATABASE\_PROJECT`.`AIRPLANE\_TYPE` (`Airplane\_type\_name`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `DATABASE\_PROJECT`.`LEG\_INSTANCE`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `DATABASE\_PROJECT`.`LEG\_INSTANCE` (

`Flight\_number` INT NOT NULL,

`Leg\_number` INT NOT NULL,

`Date` DATE NOT NULL,

`Number\_of\_available\_seats` INT NULL,

`Airplane\_id` INT NULL,

`Departure\_airport\_code` INT NOT NULL,

`Arrival\_airport\_code` INT NOT NULL,

`Arrival\_time` DATETIME NOT NULL,

`Departure\_time` DATETIME NOT NULL,

PRIMARY KEY (`Flight\_number`, `Leg\_number`, `Date`),

INDEX `fk\_Leg\_INSTANCE\_Airplane\_id\_idx` (`Airplane\_id` ASC) VISIBLE,

INDEX `fk\_Leg\_INSTANCE\_Arrival\_airprot\_code\_idx` (`Arrival\_airport\_code` ASC) VISIBLE,

INDEX `fk\_Leg\_INSTANCE\_Departure\_airport\_code\_idx` (`Departure\_airport\_code` ASC) VISIBLE,

INDEX `fk\_LEG\_INSTANCE\_Leg\_number\_idx` (`Leg\_number` ASC) VISIBLE,

CONSTRAINT `fk\_LEG\_INSTANCE\_Flight\_number`

FOREIGN KEY (`Flight\_number`)

REFERENCES `DATABASE\_PROJECT`.`FLIGHT` (`Flight\_number`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_LEG\_INSTANCE\_Leg\_number`

FOREIGN KEY (`Leg\_number`)

REFERENCES `DATABASE\_PROJECT`.`FLIGHT\_LEG` (`Flight\_number`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_Leg\_INSTANCE\_Airplane\_id`

FOREIGN KEY (`Airplane\_id`)

REFERENCES `DATABASE\_PROJECT`.`AIRPLANE` (`Airplane\_id`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_Leg\_INSTANCE\_Arrival\_airprot\_code`

FOREIGN KEY (`Arrival\_airport\_code`)

REFERENCES `DATABASE\_PROJECT`.`AIRPORT` (`Airport\_code`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_Leg\_INSTANCE\_Departure\_airport\_code`

FOREIGN KEY (`Departure\_airport\_code`)

REFERENCES `DATABASE\_PROJECT`.`AIRPORT` (`Airport\_code`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `DATABASE\_PROJECT`.`FARE`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `DATABASE\_PROJECT`.`FARE` (

`Flight\_number` INT NOT NULL,

`Fare\_code` INT NOT NULL,

`Amount` VARCHAR(45) NOT NULL,

`Restrictions` INT NOT NULL,

PRIMARY KEY (`Flight\_number`, `Fare\_code`),

CONSTRAINT `fk\_FARE\_Flight\_number`

FOREIGN KEY (`Flight\_number`)

REFERENCES `DATABASE\_PROJECT`.`FLIGHT` (`Flight\_number`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `DATABASE\_PROJECT`.`CAN\_LAND`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `DATABASE\_PROJECT`.`CAN\_LAND` (

`Airplane\_type\_name` VARCHAR(15) NOT NULL,

`Airport\_code` INT NOT NULL,

PRIMARY KEY (`Airplane\_type\_name`, `Airport\_code`),

INDEX `fk\_CAN\_LAND\_Airport\_code\_idx` (`Airport\_code` ASC) VISIBLE,

CONSTRAINT `fk\_CAN\_LAND\_Airplane\_type\_name`

FOREIGN KEY (`Airplane\_type\_name`)

REFERENCES `DATABASE\_PROJECT`.`AIRPLANE\_TYPE` (`Airplane\_type\_name`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_CAN\_LAND\_Airport\_code`

FOREIGN KEY (`Airport\_code`)

REFERENCES `DATABASE\_PROJECT`.`AIRPORT` (`Airport\_code`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `DATABASE\_PROJECT`.`SEAT\_RESERVATION`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `DATABASE\_PROJECT`.`SEAT\_RESERVATION` (

`Flight\_number` INT NOT NULL,

`Leg\_number` INT NOT NULL,

`Date` DATE NOT NULL,

`Seat\_number` VARCHAR(45) NOT NULL,

`Customer\_name` VARCHAR(20) NULL,

`Customer\_phone` VARCHAR(20) NULL,

PRIMARY KEY (`Flight\_number`, `Leg\_number`, `Date`, `Seat\_number`),

INDEX `fk\_SEAT\_RESERVATION\_Leg\_number\_idx` (`Leg\_number` ASC) VISIBLE,

CONSTRAINT `fk\_SEAT\_RESERVATION\_Leg\_number`

FOREIGN KEY (`Leg\_number`)

REFERENCES `DATABASE\_PROJECT`.`FLIGHT\_LEG` (`Flight\_number`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_SEAT\_RESERVATION\_Date`

FOREIGN KEY (`Flight\_number`)

REFERENCES `DATABASE\_PROJECT`.`FLIGHT` (`Flight\_number`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_SEAT\_RESERVATION\_Flight\_number`

FOREIGN KEY (`Leg\_number`)

REFERENCES `DATABASE\_PROJECT`.`FLIGHT\_LEG` (`Flight\_number`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `DATABASE\_PROJECT`.`USER`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `DATABASE\_PROJECT`.`USER` (

`UserID` INT NOT NULL,

`Flight\_number` INT NOT NULL,

`Date` DATE NOT NULL,

`Seat\_number` VARCHAR(45) NOT NULL,

`Customer\_name` VARCHAR(20) NOT NULL,

`Customer\_phone` VARCHAR(20) NOT NULL,

PRIMARY KEY (`UserID`, `Flight\_number`),

INDEX `fk\_USER\_Flight\_number\_idx` (`Flight\_number` ASC) VISIBLE,

CONSTRAINT `fk\_USER\_Flight\_number`

FOREIGN KEY (`Flight\_number`)

REFERENCES `DATABASE\_PROJECT`.`FLIGHT` (`Flight\_number`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

Flight table의 Weekdays와 를 삭제하였습니다.

텍스트이(가) 표시된 사진

자동 생성된 설명

<CREATE INDEX설정>

Flight Table 튜플 값 입력

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

텍스트이(가) 표시된 사진

자동 생성된 설명

다음은 각 테이블마다 인덱스르 3~4개씩 설정하였습니다. 인덱스가 효율적인 검색에 도움을 주어 필수적인 키의 인덱스를 추가하였지만 무조건 많이 인덱스를 설정한다고 해서 검색 속도를 향상시키지는 않으므로 최대한 중복이 되지 않게 인덱스를 설정해 주었습니다. 특히 WHERE에 자주 사용되는 컬럼에 주로 사용하거나 JOIN에 자주 사용되는 컬럼에는 인덱스를 많이 추가하였으며 데이터의 변경이 잦은 컬럼에는 인덱스를 사용하지 않았습니다.

4. 실행 화면

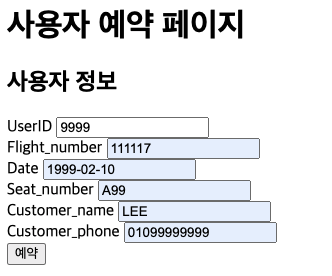
1.



첫 실행 화면은 <http://localhost:3000/> 에서 사용자가 아이디와 비밀번호를 입력하여 항공편을 예약할 수 있는 페이지를 넘어갈 수 있도록 구성을 하였습니다.

2.

<http://localhost:3000/res_user> 로그인이 성공한 페이지는 다음과 같이 사용자 예약페이지로 이동하며 사용자의 개인 정보, 공항 정보, 항공기 정보, 항공편 정보를 입력할 수 있도록 구성하였습니다.



이어서 예약한 USERID, Flight\_number, Date, Seat\_number, Cutomer\_name, Cusomer\_phone 등 user table에 insert하여 저장되고 <http://localhost:3000/select> 에 정보가 저장될 수 있도록 설계하였습니다.

테이블이(가) 표시된 사진

자동 생성된 설명

http://localhost:3000/select

다음과 같이 사용자 예약 조회 페이지는 USERID, Flight\_number, Date, Seat\_number, Customer\_name, Customer\_phone등이 저장이 되는 것을 확인할 수 있었습니다.

테이블이(가) 표시된 사진

자동 생성된 설명

<http://localhost:3000/delete/user>

취소 페이지는 기본에 저장되어 있는 테이블 옆에 ‘삭제’라는 버튼을 형성하게 하였고 버튼을 누르면 해당하는 튜플이 사라지도록 설계하였습니다.

테이블이(가) 표시된 사진

자동 생성된 설명

튜플 중 잘못 기입된 번호를 삭제 버튼 눌러 정상적으로 작동함을 보였습니다.

3.

텍스트이(가) 표시된 사진

자동 생성된 설명

다음은 관리자페이지로 관리자페이지도 사용자 정보가 필요하다고 생각되어 사용자 정보, 공항 정보, 항공기 정보, 항공편 정보 등을 포함하여 구성하였습니다. 사용자페이지에서 예약하기와도 중복되므로 삭제를 하고 진행하기도 하였습니다.

테이블이(가) 표시된 사진

자동 생성된 설명

관리자 삽입페이지에서 정보를 추가하여 테이블 추가한 모습이며 삭제 기능을 할 수 있는 페이지입니다. Mysql에서 insert문을 추가하였을 때도 값이 정상적으로 입력되고 삽입페이지를 통해서 값을 삽입하더라도 정상적으로 작동하였습니다. 공항 정보, 항공기 정보, 항공편 정보는 서로 각각 다른 페이지를 만든것이 아니라 관리자가 쉽게 관리 할 수 있도록 테이블을 모아서 구성하였습니다.

테이블이(가) 표시된 사진

자동 생성된 설명

다음은 관리자 수정 페이지입니다. 수정페이지는 삭제페이지와는 달리 값이 완전하게 사라지는 것이 아니라 테이블을 유지한 채 사용자의 임의대로 수정할 수 있는 기능을 가지고 있습니다. 현재 페이지에서는 Airport\_code, Airplane\_id, Flight\_number 만 수정이 가능한 기능 만을 추가하였지만 사용자의 입장에 따라서 유동적으로 변경이 가능할 것 같습니다.

고찰

이번 프로젝트를 하면서 처음 설계부터 실제로 웹으로 결과물을 보면서 조금 더 열심히 하면 완성도가 높은 설계가 될 수 있을것 이라고 욕심을 가지게 되었던 것 같습니다. 아쉬운 점은 트랜젹션 설계와 정규화 작업에 대하여 이론은 파악하다고 생각하였으나 실습에서는 제대로 이루어졌는지 확실치 못한 점이 아쉬웠습니다. 그 외에 제약조건과 성능같은 경우 최대한 구현하려고 노력하였으나 마지막 수정하는 부분에서 코드에 오류가 발생하였는지 완벽한 구현은 하지 못했던 것 같습니다.

프로젝트를 하면서 직접 설계하는 시간보다는 보다 효율적으로 설계하기 위해 여러 종류의 ER설계 중 어떤 것이 후에 관리하기가 설계할 것인가 또 필요한 스키마와 설계 구성에 대해서 많이 고민하고 좋은 설계란 무엇인가에 대해 생각했던 것 같습니다. 처음 설계를 시작할 때는 실무에서 사용할 수 있도록 여러 기능을 가진 데이터베이스 설계를 하려 했지만 기능이 더해질수록 고려해야할 점이 많고 특히 제약조건에서 여러가지 걸림돌이 되어 다시 처음부터 시작하여 수업 실습 위주로의 설계를 하였습니다. 설계 중에서도 도서관 도서 대출, 회원 관리 등등 여러 데이터베이스가 있지만 AIRLINE데이터베이스는 실생활에서도 접할 수 있으면서도 생각보다 도전적인 데이터베이스 설계라서 실력향상에는 더 좋았던 것 같습니다.