Enjoy Trip Project

DB Modeling

1. 개요

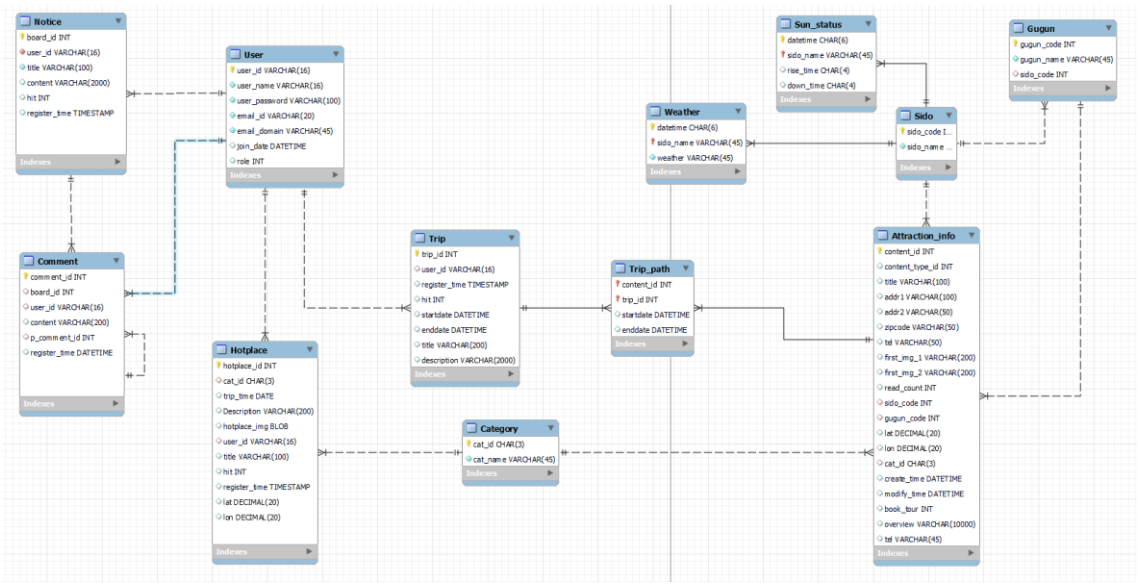
- PJT명 : Enjoy Trip Project
- ·단계 : DB Modeling PJT
- 진행일자 : 2023.04.14
- 예상 구현 시간 :
- 필수기능 8H, 추가기능 1H, 심화기능 2H
- 팀원 : 최재용, 황제원
- 개발 언어 / 프로그램
  - Java / Eclipse
  - MySQL / MySQL Workbench / ERD 관련

2. 요구 사항

사용자에게 한국의 다양한 관광지, 먹거리, 축제, 행사 등을 소개하여 지역 관광 활성화를 위한 지역 관광 소개 페이지를 구축하려고 한다.
한국관광공사에서 제공하는 국문 관광정보서비스_GW의 다양한 상세기능정보 API를 활용하여 지역별 관광지 data를 분석하고 화면에 표시한다. 또한 여행계획을 위한 스케줄과 여행경로 공유 등 사용자 편의 기능을 구현하고 나만의 숨은 관광지를 소개하는 페이지와 자유롭게 토론이 가능한 게시판 등을 구현해 본다.

추가적으로 관광지의 날씨정보나 일출, 일몰시각, 관광경로상의 전기차 충전소등 다양한 아이디어를 통한 추가 기능도 구현해 보자.
위 기능들을 화면에 표시하기위해 필요한 Database를 설계하고 구현하여 보자.

3. ERD



[ERD 개요]

저희는 데이터베이스 엔티티로 Notice(공지사항), User(사용자), Comment(댓글), Hotplace(핫플레이스), Trip(여행계획), Trip_path(여행계획 상 여행지), Category(관광지 유형), Weather(지역 날씨), Sun_status(일출/일몰), Sido(시), Gugun(구), Attraction_info(관광지)를 정의했습니다.

[DDL – Forward Engineering 사용]

4. DDL

1. User(사용자)

```
-- -----------------------------------------------------
-- Table `mydb`.`User`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`User` (
  `user_id` VARCHAR(16) NOT NULL,
  `user_name` VARCHAR(16) NOT NULL,
  `user_password` VARCHAR(100) NOT NULL,
  `email_id` VARCHAR(20) NOT NULL,
  `email_domain` VARCHAR(45) NOT NULL,
  `join_date` DATETIME NULL DEFAULT now(),
  `role` INT NULL,
  PRIMARY KEY (`user_id`),
  UNIQUE INDEX `email_id_UNIQUE` (`email_id` ASC) VISIBLE)
ENGINE = InnoDB;
```

2. Notice(공지사항)

```
-- -----------------------------------------------------
-- Table `mydb`.`Notice`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`Notice` (
  `board_id` INT NOT NULL AUTO_INCREMENT,
  `user_id` VARCHAR(16) NOT NULL,
  `title` VARCHAR(100) NOT NULL,
  `content` VARCHAR(2000) NULL,
  `hit` INT NULL,
  `register_time` TIMESTAMP NULL,
  PRIMARY KEY (`board_id`),
  INDEX `notice_to_member_user_id_fk_idx` (`user_id` ASC) VISIBLE,
  CONSTRAINT `notice_to_member_user_id_fk`
    FOREIGN KEY (`user_id`)
    REFERENCES `mydb`.`User` (`user_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

3. Category(카테고리)

```
-- -----------------------------------------------------
-- Table `mydb`.`Category`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`Category` (
  `cat_id` CHAR(3) NOT NULL,
  `cat_name` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`cat_id`))
ENGINE = InnoDB;
```

4. Hotplace(핫플레이스)

```
-- -----------------------------------------------------
-- Table `mydb`.`Hotplace`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`Hotplace` (
  `hotplace_id` INT NOT NULL AUTO_INCREMENT,
  `cat_id` CHAR(3) NULL,
  `trip_time` DATE NULL,
  `Description` VARCHAR(200) NULL,
  `hotplace_img` BLOB NULL,
  `user_id` VARCHAR(16) NULL,
  `title` VARCHAR(100) NULL,
  `hit` INT NULL DEFAULT 0,
  `register_time` TIMESTAMP NULL,
  `lat` DECIMAL(20) NULL,
  `lon` DECIMAL(20) NULL,
  PRIMARY KEY (`hotplace_id`),
  INDEX `hotplace_to_cat_cat_id_fk_idx` (`cat_id` ASC) VISIBLE,
  INDEX `hotplace_to_memer_user_id_fk_idx` (`user_id` ASC) VISIBLE,
  CONSTRAINT `hotplace_to_cat_cat_id_fk`
    FOREIGN KEY (`cat_id`)
    REFERENCES `mydb`.`Category` (`cat_id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE,
  CONSTRAINT `hotplace_to_member_user_id_fk`
    FOREIGN KEY (`user_id`)
    REFERENCES `mydb`.`User` (`user_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

5. Trip_path(여행계획)

```
-- -----------------------------------------------------
-- Table `mydb`.`Trip_path`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`Trip_path` (
  `trip_path_id` INT NOT NULL AUTO_INCREMENT,
  `user_id` VARCHAR(16) NULL,
  `register_time` TIMESTAMP NULL,
  `hit` INT NULL,
  `startdate` DATETIME NULL,
  `enddate` DATETIME NULL,
  `title` VARCHAR(200) NULL,
  `description` VARCHAR(2000) NULL,
  PRIMARY KEY (`trip_path_id`),
  INDEX `path_to_user_id_fk_idx` (`user_id` ASC) VISIBLE,
  CONSTRAINT `path_to_user_id_fk`
    FOREIGN KEY (`user_id`)
    REFERENCES `mydb`.`User` (`user_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

6. Sido(시)

```
-- -----------------------------------------------------
-- Table `mydb`.`Sido`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`Sido` (
  `sido_code` INT NOT NULL,
  `sido_name` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`sido_code`),
  UNIQUE INDEX `sido_name_UNIQUE` (`sido_name` ASC) VISIBLE)
ENGINE = InnoDB;
```

7. Gugun(구)

```
-- -----------------------------------------------------
-- Table `mydb`.`Gugun`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`Gugun` (
  `gugun_code` INT NOT NULL,
  `gugun_name` VARCHAR(45) NOT NULL,
  `sido_code` INT NULL,
  PRIMARY KEY (`gugun_code`),
  INDEX `gugun_to_sido_code_fk_idx` (`sido_code` ASC) VISIBLE,
  CONSTRAINT `gugun_to_sido_code_fk`
    FOREIGN KEY (`sido_code`)
    REFERENCES `mydb`.`Sido` (`sido_code`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

8. Attraction_Info(관광지)

```sql
-- -----------------------------------------------------
-- Table `mydb`.`Attraction_info`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`Attraction_info` (
  `content_id` INT NOT NULL,
  `content_type_id` INT NULL,
  `title` VARCHAR(100) NULL,
  `addr1` VARCHAR(100) NULL,
  `addr2` VARCHAR(50) NULL,
  `zipcode` VARCHAR(50) NULL,
  `tel` VARCHAR(50) NULL,
  `first_img_1` VARCHAR(200) NULL,
  `first_img_2` VARCHAR(200) NULL,
  `read_count` INT NULL,
  `sido_code` INT NULL,
  `gugun_code` INT NULL,
  `lat` DECIMAL(20) NULL,
  `lon` DECIMAL(20) NULL,
  `cat_id` CHAR(3) NULL,
  `create_time` DATETIME NULL,
  `modify_time` DATETIME NULL,
  `book_tour` INT NULL,
  `overview` VARCHAR(10000) NULL,
  `tel` VARCHAR(45) NULL,
  PRIMARY KEY (`content_id`),
  INDEX `info_to_sido_code_fk_idx_idx` (`sido_code` ASC) VISIBLE,
  INDEX `info_to_gugun_code_fk_idx_idx` (`gugun_code` ASC) VISIBLE,
  INDEX `info_to_cat_cat_id_fk_idx` (`cat_id` ASC) VISIBLE,
  CONSTRAINT `info_to_sido_code_fk_idx`
    FOREIGN KEY (`sido_code`)
    REFERENCES `mydb`.`Sido` (`sido_code`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `info_to_gugun_code_fk_idx`
    FOREIGN KEY (`gugun_code`)
    REFERENCES `mydb`.`Gugun` (`gugun_code`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `info_to_cat_cat_id_fk`
    FOREIGN KEY (`cat_id`)
    REFERENCES `mydb`.`Category` (`cat_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

9. Weather(날씨)

```
-- -----------------------------------------------------
-- Table `mydb`.`Weather`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`Weather` (
  `datetime` CHAR(6) NOT NULL,
  `sido_name` VARCHAR(45) NOT NULL,
  `weather` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`datetime`, `sido_name`),
  INDEX `weather_to_sido_idx` (`sido_name` ASC) VISIBLE,
  CONSTRAINT `weather_to_sido`
    FOREIGN KEY (`sido_name`)
    REFERENCES `mydb`.`Sido` (`sido_name`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

10. Sun_status (일출/일몰)

```
-- -----------------------------------------------------
-- Table `mydb`.`Sun_status`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`Sun_status` (
  `datetime` CHAR(6) NOT NULL,
  `sido_name` VARCHAR(45) NOT NULL,
  `rise_time` CHAR(4) NULL,
  `down_time` CHAR(4) NULL,
  PRIMARY KEY (`datetime`, `sido_name`),
  INDEX `sun_to_sido_fk_idx` (`sido_name` ASC) VISIBLE,
  CONSTRAINT `sun_to_sido_fk`
    FOREIGN KEY (`sido_name`)
    REFERENCES `mydb`.`Sido` (`sido_name`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

11. Comment (댓글)

```sql
-- -----------------------------------------------------
-- Table `mydb`.`Comment`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`Comment` (
  `comment_id` INT NOT NULL AUTO_INCREMENT,
  `board_id` INT NULL,
  `user_id` VARCHAR(16) NULL,
  `content` VARCHAR(200) NULL,
  `p_comment_id` INT NULL,
  `register_time` DATETIME NULL,
  PRIMARY KEY (`comment_id`),
  INDEX `comment_to_user_id_fk_idx` (`user_id` ASC) VISIBLE,
  INDEX `comment_to_comment_fk_idx` (`p_comment_id` ASC) VISIBLE,
  INDEX `comment_to_notice_board_id_fk_idx` (`board_id` ASC) VISIBLE,
  CONSTRAINT `comment_to_comment_fk`
    FOREIGN KEY (`p_comment_id`)
    REFERENCES `mydb`.`Comment` (`comment_id`)
    ON DELETE CASCADE
    ON UPDATE NO ACTION,
  CONSTRAINT `comment_to_user_id_fk`
    FOREIGN KEY (`user_id`)
    REFERENCES `mydb`.`User` (`user_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `comment_to_notice_board_id_fk`
    FOREIGN KEY (`board_id`)
    REFERENCES `mydb`.`Notice` (`board_id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB;
```

12. Trip_path_location (여행계획 상 여행지)

```sql
-- -----------------------------------------------------
-- Table `mydb`.`Trip_path_location`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`Trip_path_location` (
  `content_id` INT NOT NULL,
  `trip_path_id` INT NOT NULL,
  `startdate` DATETIME NULL,
  `enddate` DATETIME NULL,
  PRIMARY KEY (`content_id`, `trip_path_id`),
  INDEX `location_to_trip_path_id_fk_idx` (`trip_path_id` ASC) VISIBLE,
  CONSTRAINT `location_to_trip_path_id_fk`
    FOREIGN KEY (`trip_path_id`)
    REFERENCES `mydb`.`Trip_path` (`trip_path_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `location_to_attraction_info`
    FOREIGN KEY (`content_id`)
    REFERENCES `mydb`.`Attraction_info` (`content_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

5. DAO

1. AttractionDAO

```
package com.ssafy.dao;

import java.sql.SQLException;
import java.util.List;
import java.util.Map;

import com.ssafy.dto.AttractionDto;

public interface AttractionDao {
	// 글 업로드
	void writeAttraction(AttractionDto attractionDto) throws SQLException;

	// 정보 목록 조회
	List<AttractionDto> listAttraction() throws SQLException;

	// 정보 조회
	AttractionDto getAttraction(int content_id) throws SQLException;

	// 정보 수정
	void modifyAttraction(AttractionDto attractionDto) throws SQLException;

	// 정보 삭제
	void deleteAttraction(int content_id) throws SQLException;
}
```

2. HotplaceBoardDao

```java
package com.ssafy.dao;

import java.sql.SQLException;
import java.util.List;
import java.util.Map;

import com.ssafy.dto.HotplaceDto;

// HotplaceDto
public interface HotplaceBoardDao {
	// 글쓰기
	void writeHotplace(HotplaceDto hotplaceDto) throws SQLException;

	// 글 목록 조회
	List<HotplaceDto> listHotplace() throws SQLException;

	// 글 조회
	HotplaceDto getHotplace(int hotplace_id) throws SQLException;

	// 조회수 업데이트
	void updateHit(int hotplace_id) throws SQLException;

	// 게시글 수정
	void modifyHotplace(HotplaceDto hotplaceDto) throws SQLException;

	// 게시글 삭제
	void deleteHotplace(int hotplace_id) throws SQLException;
}
```

3. NoticeBoardDao

```java
package com.ssafy.dao;

import java.sql.SQLException;
import java.util.List;
import java.util.Map;

import com.ssafy.dto.NoticeDto;

public interface NoticeBoardDao {

	// 글쓰기
	void writeNotice(NoticeDto noticeDto) throws SQLException;

	// 글 목록 조회
	List<NoticeDto> listNotice() throws SQLException;

	// 글 조회
	NoticeDto getNotice(int board_id) throws SQLException;

	// 조회수 업데이트
	void updateHit(int board_id) throws SQLException;

	// 게시글 수정
	void modifyNotice(NoticeDto noticeDto) throws SQLException;

	// 게시글 삭제
	void deleteNotice(int board_id) throws SQLException;
}
```

4. NoticeCommentDao

```java
package com.ssafy.dao;

import java.sql.SQLException;
import java.util.List;

import com.ssafy.dto.NoticeCommentDto;

public interface NoticeCommentDao {

	// 댓글쓰기
	// 대댓글 쓰기
	void writeComment(NoticeCommentDto noticeCommentDto) throws SQLException;

	// 댓글 목록 조회
	List<NoticeCommentDto> listComment() throws SQLException;

	// 댓글 조회
	NoticeCommentDto getComment(int comment_id) throws SQLException;

	// 댓글 수정
	void modifyComment(NoticeCommentDto noticeCommentDto) throws SQLException;

	// 댓글 삭제
	void deleteComment(int comment_id) throws SQLException;
}
```

5. TripBoardDao

```java
package com.ssafy.dao;

import java.sql.SQLException;
import java.util.List;

import com.ssafy.dto.HotplaceDto;
import com.ssafy.dto.TripBoardDto;

public interface TripBoardDao {
	// 글쓰기
	void writeHotplace(TripBoardDto tripBoardDto) throws SQLException;

	// 글 목록 조회
	List<TripBoardDto> listHotplace() throws SQLException;

	// 글 조회
	TripBoardDto getHotplace(int trip_id) throws SQLException;

	// 조회수 업데이트
	void updateHit(int trip_id) throws SQLException;

	// 게시글 수정
	void modifyHotplace(TripBoardDto tripBoardDto) throws SQLException;

	// 게시글 삭제
	void deleteHotplace(int trip_id) throws SQLException;
}
```

6. UserDao

```java
package com.ssafy.dao;

import java.sql.SQLException;

import com.ssafy.dto.UserDto;

public interface UserDao {
	int idCheck(String userId) throws SQLException;

	// 회원가입
	int joinMember(UserDto userDto) throws SQLException;

	// 로그인
	UserDto loginMember(String userId, String userPwd) throws SQLException;

	// 아이디 수정
	int update(String userId, String userPwd) throws SQLException;

	// 회원 탈퇴
	int delete(String userId) throws SQLException;
}
```