



# Team7 DBISFREE

## 목차

- [1. 해당 데이터베이스 필요성 설명 및 요구분석](#)
- [2. ER 다이어그램](#)
- [3. 데이터베이스 스키마 다이어그램](#)
- [4. Java 코드 설명](#)
- [5. 응용프로그램 사용 방법](#)
- [6. 요구사항 분석](#)
- [7. 요구조건 외의 팀만의 강점](#)
- [8. 팀 구성원 담당 부분](#)

## 팀원

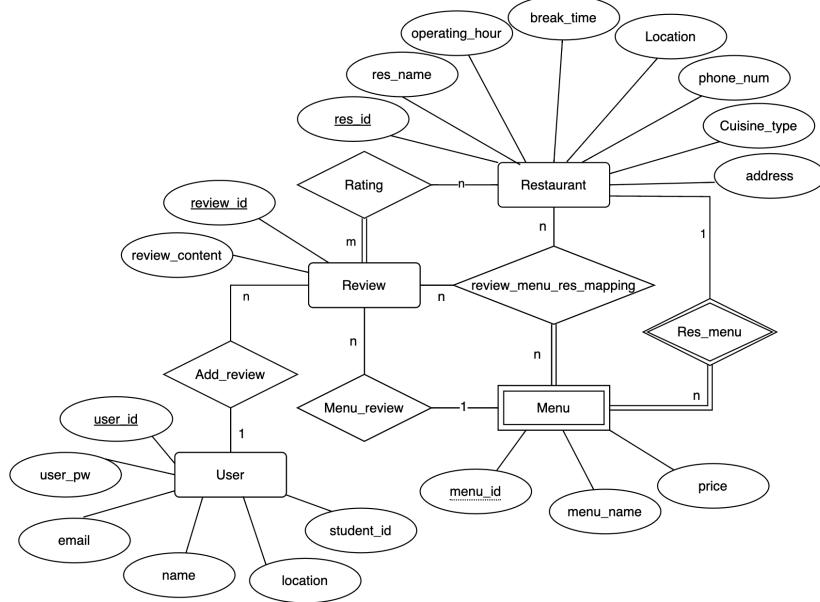
고은서 2122004  
김민서 2276046  
조서정 2276305  
차현주 2276321  
한사랑 2271064

## ▼ 1. 해당 데이터베이스 필요성 설명 및 요구분석

**E-MATEASY** - 이대생들을 위한 맛집 *easy*하게 찾기

- 맛집 프로그램에는 식당, 메뉴, 평점, 리뷰 등 다양한 데이터들이 서로 복잡한 관계를 맺고 있다. 이러한 데이터들을 체계적으로 관리하고 사용자에게 유용한 정보를 제공해야 한다.
- 식당, 메뉴, 평점, 리뷰 등의 데이터를 통합적으로 관리함으로써 데이터의 일관성과 무결성을 유지할 수 있다. 이를 통해 사용자는 최신 정보를 제공받고, 관리자는 효율적으로 데이터를 처리할 수 있다.

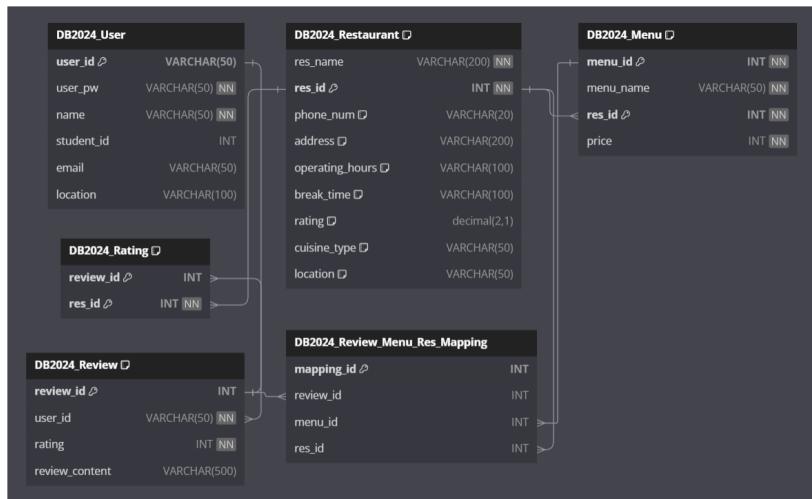
## ▼ 2. ER 다이어그램



- 총 4개의 Entity

- DB2024\_User: E-MATEASY를 사용하는 유저들의 정보
- DB2024\_Menu: E-MATEASY에 등록되어있는 메뉴들의 정보
- DB2024\_Restaurant: E-MATEASY에 등록되어 있는 식당들의 정보
- DB2024\_Review: E-MATEASY에 등록되어 있는 리뷰들의 정보

### ▼ 3. 데이터베이스 스키마 디아이그램



- User\_Review

: 각 리뷰는 한 사용자가 작성하고 한 명의 사용자는 여러개의 리뷰를 등록할 수 있으므로 DB2024\_Review와 DB2024\_User는 일대다(One-to-Many) 관계를 맺는다.

- Restaurant\_Menu

: 하나의 메뉴는 한개의 식당에 존재하고 하나의 식당은 여러 개의 메뉴를 가질 수 있으므로 DB2024\_Restaurant와 DB2024\_Menu는 일대다(One-to-Many) 관계이다.

- Restaurant\_Review

: 하나의 리뷰는 한 식당에 대한 리뷰이고 하나의 식당은 여러 개의 리뷰를 가질 수 있으므로 DB2024\_Restaurant와 DB2024\_Review는 일대다(One-to-Many) 관계이다.

- **Menu\_Review**

: 하나의 메뉴에 여러개의 리뷰가 존재하고 또 하나의 리뷰는 한 메뉴에 대한 리뷰이므로, DB2024\_Menu와 DB2024\_Review는 일대다(One-to-Many) 관계이다.

- **Review\_Menu\_Res\_Mapping**

: 여러 식당에 여러 메뉴에 대한 여러 리뷰가 존재할 수 있으므로 DB2024\_Review와 DB2024\_Menu, 그리고 DB2024\_Restaurant 간의 3진 다대다(Many-to-Many) 관계를 가진다.

## ▼ 4. Java 코드 설명

### 전체 프로젝트 구조

```

src
└── com
    └── app
        ├── DB2024TEAM07_Main           # 프로그램 실행 진입점
        ├── DB2024TEAM07_UserMain       # 유저 메인 화면
        └── DB2024TEAM07_AdminMain      # 관리자 메인 화면
    └── jdbc
        └── database
            ├── DB2024TEAM07_Database     # 데이터베이스 연결 및 공통 작업 처리
            ├── DB2024TEAM07_UserDAO       # User 테이블 CRUD 작업 처리
            ├── DB2024TEAM07_RestaurantDAO # Restaurant 테이블 CRUD 작업 처리
            ├── DB2024TEAM07_ReviewDAO     # Review 테이블 CRUD 작업 처리
            └── DB2024TEAM07_MenuDAO       # Menu 테이블 CRUD 작업 처리
        └── model
            ├── DB2024TEAM07_User          # 모델 클래스 (DTO)
            ├── DB2024TEAM07_Restaurant    # 모델 클래스 (DTO)
            ├── DB2024TEAM07_Review         # 모델 클래스 (DTO)
            └── DB2024TEAM07_Menu          # 모델 클래스 (DTO)
        └── view
            ├── DB2024TEAM07_ResReviewVO   # Restaurant, Review 뷰 클래스
            └── DB2024TEAM07_UserVO         # User 뷰 클래스
    └── manager
        ├── DB2024TEAM07_RestaurantManager # 식당 관련 비즈니스 로직 처리
        ├── DB2024TEAM07_MenuManager       # 메뉴 관련 비즈니스 로직 처리
        ├── DB2024TEAM07_ReviewManager     # 리뷰 관련 비즈니스 로직 처리
        └── DB2024TEAM07_UserManager       # 유저 관련 비즈니스 로직 처리

```

### 클래스 및 메소드 상세 설명

#### ▼ package com.app;

Classes	
Class	Description
DB2024TEAM07_AdminMain	Main class for administrator functionalities in the E-MATEASY application.
DB2024TEAM07_Main	This class represents the main entry point of the application.
DB2024TEAM07_UserMain	User interface for logged-in users.

#### ▼ Main

: 사용자에게 메뉴 기반 인터페이스를 제공하여 시스템과 상호작용할 수 있게 하는 메인 클래스이다.

```

package com.app;

import com.manager.DB2024TEAM07_UserManager;
import com.jdbc.database.DB2024TEAM07_UserDAO;

```

```

import com.jdbc.model.DB2024TEAM07_User;

import java.util.Scanner;

/**
 * This class represents the main entry point of the application.
 * It provides a menu-driven interface for users to interact with the system.
 *
 */
public class DB2024TEAM07_Main {
    private static final String ADMIN_PASSWORD = "admin123";

    public static void main(String[] args) {
        DB2024TEAM07_UserDAO userDAO = new DB2024TEAM07_UserDAO();
        DB2024TEAM07_UserManager userManager = new DB2024TEAM07_UserManager(userDAO);
        Scanner scanner = new Scanner(System.in);

        while (true) {
            System.out.println(" ");
            System.out.println(" E-MATEASY ");
            System.out.println("\n=====\\n");
            System.out.println("1. User Login");
            System.out.println("2. User Register");
            System.out.println("3. Admin Login");
            System.out.println("4. Exit");
            System.out.println("\n=====");

            boolean validChoice = false;
            int choice = 0;

            while (!validChoice) {
                System.out.print("Choose an option: ");
                try {
                    choice = scanner.nextInt();
                    scanner.nextLine();

                    if (choice >= 1 && choice <= 4) {
                        validChoice = true;
                    } else {
                        System.out.println("Invalid choice. Please enter a number between 1 and 4.");
                    }
                } catch (Exception e) {
                    System.out.println("Invalid input. Please enter a number.");
                    scanner.nextLine();
                }
            }

            switch (choice) {
                case 1:
                    // User Login
                    System.out.print("Enter username: ");
                    String username = scanner.nextLine();

                    System.out.print("Enter password: ");
                    String password = scanner.nextLine();

                    if (userManager.login(username, password)) {
                        System.out.println("Login successful!");
                    }
            }
        }
    }
}

```

```

        DB2024TEAM07_UserMain.showMenu(userManager, username);
    } else {
        System.out.println("Invalid username or password.");
    }
    break;
case 2:
    // User Register
    System.out.print("New username: ");
    String newUsername = scanner.nextLine();

    // Validate username format (maximum length of 50 characters)
    if (newUsername.length() > 50) {
        System.out.println("Username cannot exceed 50 characters.");
        return;
    }

    System.out.print("New password: ");
    String newPassword = scanner.nextLine();

    // Validate password format (maximum length of 50 characters)
    if (newPassword.length() > 50) {
        System.out.println("Password cannot exceed 50 characters.");
        return;
    }

    System.out.print("Name: ");
    String name = scanner.nextLine();

    if (name.length() > 50) {
        System.out.println("Name cannot exceed 50 characters.");
        return;
    }

    System.out.print("Student ID: ");
    String studentIdStr = scanner.nextLine();

    while (!studentIdStr.matches("[0-9]{1,50}")) {
        System.out.println("Invalid student ID. Please enter a numerical value between 1 and 50 characters long.");
        studentIdStr = scanner.nextLine();
    }

    int studentId = Integer.parseInt(studentIdStr);

    String email;
    do {
        System.out.print("Email (format: example@domain.com): ");
        email = scanner.nextLine();

        // Check if email follows the pattern of containing '@' and at least one '.'
        // after '@' with characters on both sides of '@'
        if (!email.matches("^[^@\s]+@[^@\s]+\.\[^@\s]+$")) {
            System.out.println("Please enter a valid email address (e.g: example@domain.com).");
        } else if (email.length() > 50) {
            System.out.println("Email cannot exceed 50 characters.");
            break;
        }
    }
}

```

```

        } while (!email.matches("^[^@\\s]+@[^@\\s]+\\. [^@\\s]+$"));

        System.out.println("1. 정문");
        System.out.println("2. 후문");
        System.out.print("Choose an Location (Enter 1 or 2): ");
        String locationChoice = scanner.nextLine();

        String location;
        switch (locationChoice) {
            case "1":
                location = "정문";
                break;
            case "2":
                location = "후문";
                break;
            default:
                System.out.println("Invalid choice. Please enter 1 for 정
문 or 2 for 후문:");
                return;
        }

        DB2024TEAM07_User newUser = new DB2024TEAM07_User(newUsername, ne
wPassword, name, studentId, email, location);
        if (userManager.addUser(newUser)) {
            System.out.println("Registration successful!");
        } else {
            System.out.println("Registration failed.");
        }
        break;
    case 3:
        // Admin login
        System.out.print("Enter admin password: ");
        String adminPassword = scanner.nextLine();

        if (ADMIN_PASSWORD.equals(adminPassword)) {
            System.out.println("Administrator login successful!");
            DB2024TEAM07_AdminMain.main(new String[] {});
        } else {
            System.out.println("Incorrect administrator password.");
        }
        break;
    case 4:
        return;
    }
}
}
}

```

- main(String[] args)
  - 프로그램의 진입점이다. 사용자가 메뉴에서 선택한 옵션에 따라 적절한 기능을 호출한다.
  - 작동 방식:
    1. `DB2024TEAM07_UserDAO` 객체를 생성하여 데이터베이스와 상호작용한다.
    2. `DB2024TEAM07_UserManager` 객체를 생성하여 사용자 관리 기능을 제공한다.
    3. `Scanner` 객체를 사용하여 사용자의 입력을 읽는다.
    4. 사용자가 메뉴에서 선택할 수 있는 옵션을 출력한다:
      - 1: 사용자 로그인

- 2: 사용자 등록
  - 3: 관리자 로그인
  - 4: 종료
5. 사용자의 선택을 기다리며 올바른 입력이 들어올 때까지 반복한다.
6. 사용자가 선택한 옵션에 따라 다음 기능을 수행한다:
- **사용자 로그인:** 사용자 이름과 비밀번호를 입력받아 로그인 시도. 성공 시 사용자 메뉴로 이동.
  - **사용자 등록:** 새로운 사용자 정보를 입력받아 등록 시도. 각 입력값에 대해 유효성 검사를 수행.
  - **관리자 로그인:** 관리자 비밀번호를 입력받아 로그인 시도. 성공 시 관리자 메뉴로 이동.
  - **종료:** 프로그램을 종료한다.

## ▼ User Main

: 로그인한 사용자를 위한 인터페이스를 제공하는 클래스이다. 레스토랑 검색, 랜덤 레스토랑 추천, 리뷰 작성, 사용자 정보 관리 및 로그아웃 등의 기능을 제공한다.

```
public static void showMenu(DB2024TEAM07_UserManager userManager, String loggedInUserName) {
    Scanner scanner = new Scanner(System.in);
    boolean running = true;

    while (running) {
        System.out.println(" ");
        System.out.println("Welcome to E-MATEASY, " + loggedInUserName + "!");
        System.out.println("=====\\n");
        System.out.println("1. Search for restaurants");
        System.out.println("2. Get Random Restaurant");
        System.out.println("3. Posting a new review");
        System.out.println("4. My Page");
        System.out.println("5. Logout");
        System.out.println("\\n=====\\n");

        boolean validChoice = false;
        int choice = 0;

        while (!validChoice) {
            System.out.print("Choose an option: ");
            try {
                choice = scanner.nextInt();
                scanner.nextLine();

                if (choice >= 1 && choice <= 5) {
                    validChoice = true;
                } else {
                    System.out.println("Invalid choice. Please enter a number between 1 and 5.");
                }
            } catch (Exception e) {
                System.out.println("Invalid input. Please enter a number.");
                scanner.nextLine();
            }
        }

        switch (choice) {
            case 1:
                showSearchMenu();
                break;
        }
    }
}
```

```

        case 2:
            DB2024TEAM07_RestaurantManager.displayRandomRestaurant();
            break;
        case 3:
            DB2024TEAM07_RestaurantManager.displayAllRestaurants();
            System.out.println("\n===== Search menu =====");
            DB2024TEAM07_MenuManager.searchMenuByRestaurant(scanner);
            System.out.println("\n===== Add review =====");
            DB2024TEAM07_ReviewManager.addReview(scanner);
            break;
        case 4:
            showUserPage();
            break;
        case 5:
            userManager.logout();
            System.out.println("Logout successful!");
            running = false;
            break;
    }
}
}

```

- showMenu(DB2024TEAM07\_UserManager userManager, String loggedInUsername)
  - 사용자가 로그인한 후의 메뉴를 표시한다.
  - 작동 방식:
    1. 사용자의 선택을 받기 위해 `Scanner` 객체를 사용한다.
    2. 메뉴 옵션을 출력하여 사용자가 선택할 수 있도록 한다:
      - 1: 레스토랑 검색
      - 2: 랜덤 레스토랑 추천
      - 3: 새로운 리뷰 작성
      - 4: 마이 페이지
      - 5: 로그아웃
    3. 사용자의 선택이 올바를 때까지 반복하여 입력을 받는다.
    4. 사용자의 선택에 따라 적절한 기능을 호출한다:
      - **레스토랑 검색:** 레스토랑 검색 메뉴를 호출한다.
      - **랜덤 레스토랑 추천:** 랜덤 레스토랑을 추천한다.
      - **새로운 리뷰 작성:** 모든 레스토랑을 표시하고, 메뉴 검색 후 리뷰를 작성한다.
      - **마이 페이지:** 사용자 정보 관리 메뉴를 호출한다.
      - **로그아웃:** 로그아웃하고 메인 메뉴로 돌아간다.

```

private static void showSearchMenu(){
    Scanner scanner = new Scanner(System.in);

    System.out.println("\n=====\n");
    System.out.println("1. Search by Restaurant Name, Cuisine Type, Location, Minimum Rating");
    System.out.println("2. Search by Cuisine Type");
    System.out.println("3. Exit");
    System.out.println("\n=====");
    System.out.print("Choose an option: ");

```

```

int choice = 0;
boolean validChoice = false;

while (!validChoice) {
    if (scanner.hasNextInt()) {
        choice = scanner.nextInt();
        scanner.nextLine();

        if (choice >= 1 && choice <= 3) {
            validChoice = true;
        } else {
            System.out.println("Invalid choice. Please enter a number between
1 and 3.");
            System.out.print("Choose an option: ");
        }
    } else {
        System.out.println("Invalid input. Please enter a number.");
        System.out.print("Choose an option: ");
        scanner.nextLine();
    }
}

switch (choice) {
    case 1:
        DB2024TEAM07_RestaurantManager.searchRestaurant(scanner);
        showResDetail();
        break;
    case 2:
        DB2024TEAM07_RestaurantManager.displayAllRestaurants();
        DB2024TEAM07_RestaurantManager.searchRestaurantByCategory(scanner);
        showResDetail();
        break;
    case 3:
        break;
}
}

```

- showSearchMenu()
  - 레스토랑을 검색할 수 있는 메뉴를 표시하는 메서드이다.
  - 작동 방식:
    1. 사용자가 선택할 수 있는 검색 옵션을 출력한다.
    2. 사용자의 선택에 따라 다음 옵션 중 하나를 실행한다:
      - 1: 레스토랑 이름, 요리 유형, 위치, 최소 평점을 기준으로 검색한다.
      - 2: 요리 유형을 기준으로 검색한다.
      - 3: 검색 메뉴를 종료한다.

```

private static void showResDetail() {
    Scanner scanner = new Scanner(System.in);
    System.out.println("\n==== Search Detail ====\n");
    System.out.println("1. Search Restaurant Menu");
    System.out.println("2. Search Restaurant Reviews");
    System.out.println("3. Exit");
    System.out.println("\n=====");
    System.out.print("Choose an option: ");
}

```

```

int choice = 0;
boolean validChoice = false;

while (!validChoice) {
    if (scanner.hasNextInt()) {
        choice = scanner.nextInt();
        scanner.nextLine();

        if (choice >= 1 && choice <= 3) {
            validChoice = true;
        } else {
            System.out.println("Invalid choice. Please enter a number between
1 and 3.");
            System.out.print("Choose an option: ");
        }
    } else {
        System.out.println("Invalid input. Please enter a number.");
        System.out.print("Choose an option: ");
        scanner.nextLine();
    }
}

switch (choice) {
    case 1:
        DB2024TEAM07_MenuManager.searchMenuByRestaurant(scanner);
        showResDetail();
        break;
    case 2:
        DB2024TEAM07_ReviewManager.getResReview(scanner);
        showResDetail();
        break;
    case 3:
        break;
}
}

```

- showResDetail()
  - 레스토랑 검색 후 세부 옵션을 표시하는 메서드이다.
  - 작동 방식:
    1. 사용자가 선택할 수 있는 세부 검색 옵션을 출력한다.
    2. 사용자의 선택에 따라 다음 옵션 중 하나를 실행한다:
      - 1: 레스토랑 메뉴를 검색한다.
      - 2: 레스토랑 리뷰를 검색한다.
      - 3: 검색 메뉴를 종료한다.

```

private static void showUserPage() {
    Scanner scanner = new Scanner(System.in);
    boolean running = true;

    while (running) {
        System.out.println("\n===== My Page =====\n");
        System.out.println("1. My information");
        System.out.println("2. Update My information");
        System.out.println("3. Search for Other Users");
        System.out.println("4. Delete My Account");
    }
}

```

```

        System.out.println("5. Exit");
        System.out.println("\n=====");
        System.out.print("Choose an option: ");

        int choice = 0;
        boolean validChoice = false;

        while (!validChoice) {
            if (scanner.hasNextInt()) {
                choice = scanner.nextInt();
                scanner.nextLine();

                if (choice >= 1 && choice <= 5) {
                    validChoice = true;
                } else {
                    System.out.println("Invalid choice. Please enter a number between 1 and 5.");
                    System.out.print("Choose an option: ");
                }
            } else {
                System.out.println("Invalid input. Please enter a number.");
                System.out.print("Choose an option: ");
                scanner.nextLine();
            }
        }

        switch (choice) {
            case 1:
                DB2024TEAM07_UserManager.showMyInfo();
                break;
            case 2:
                DB2024TEAM07_UserManager.update(scanner);
                break;
            case 3:
                DB2024TEAM07_UserManager.searchOtherUser(scanner);
                break;
            case 4:
                DB2024TEAM07_UserManager.deleteAccount(scanner);
                running = false;
                break;
            case 5:
                running = false;
                break;
        }
    }
}

```

- showUserPage()
  - 사용자가 자신의 정보를 관리할 수 있는 마이 페이지 메뉴를 표시하는 메서드이다.
  - 작동 방식:
    1. 사용자가 선택할 수 있는 사용자 정보 관리 옵션을 출력한다.
    2. 사용자의 선택에 따라 다음 옵션 중 하나를 실행한다:
      - 1: 사용자 정보를 조회한다.
      - 2: 사용자 정보를 업데이트한다.
      - 3: 다른 사용자를 검색한다.
      - 4: 계정을 삭제한다.

- 5: 마이 페이지 메뉴를 종료한다.

## ▼ Admin Main

: 사용자가 레스토랑, 메뉴, 리뷰 및 사용자를 관리하기 위한 메뉴 기반 인터페이스를 제공하는 메인 클래스이다.

- main(String[] args)
  - 프로그램의 진입점이다. 사용자가 메뉴에서 선택한 옵션에 따라 적절한 기능을 호출한다.
  - 작동 방식:
    1. `Scanner` 객체를 사용하여 사용자의 입력을 읽는다.
    2. 사용자가 선택한 메뉴 옵션에 따라 해당 기능을 호출한다.
    3. 사용자가 프로그램을 종료할 때까지 무한 반복한다.

```
package com.app;

import com.jdbc.database.DB2024TEAM07_MenuDAO;
import com.jdbc.database.DB2024TEAM07_RestaurantDAO;
import com.jdbc.database.DB2024TEAM07_UserDAO;
import com.manager.DB2024TEAM07_MenuManager;
import com.manager.DB2024TEAM07_RestaurantManager;
import com.manager.DB2024TEAM07_ReviewManager;
import com.manager.DB2024TEAM07_UserManager;

import java.util.Scanner;

import static com.manager.DB2024TEAM07_RestaurantManager.displayAllRestaurants;

/**
 * Main class for administrator functionalities in the E-MATEASY application.
 *
 * This class provides a menu-driven interface for managing restaurants, menus, reviews,
 * and users.
 */
public class DB2024TEAM07_AdminMain {

    /**
     * The main entry point of the application.
     *
     * @param args command line arguments (unused in this program)
     */
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        DB2024TEAM07_UserDAO userDAO = new DB2024TEAM07_UserDAO();
        DB2024TEAM07_UserManager userManager = new DB2024TEAM07_UserManager(userDAO);

        while (true) {
            System.out.println("\n===== Restaurant ====== ====== Menu =====\n");
            System.out.println(" 1. Add Restaurant | 5. Add Menu");
            System.out.println(" 2. Update Restaurant | 6. Update Menu");
            System.out.println(" 3. Search Restaurant | 7. Search Menu");
            System.out.println(" 4. Delete Restaurant | 8. Delete Menu");
            System.out.println("\n===== Review ====== ====== User =====\n");
            System.out.println(" 9. Add Review | 13. Add User");
            System.out.println(" 10. Update Review | 14. Update User");
            System.out.println(" 11. Search Review | 15. Search User");
        }
    }
}
```

```

        System.out.println(" 12. Delete Review      |   16. Delete User");
        System.out.println("\n===== =====");
====\n");
        System.out.println("17. Exit\n");
        System.out.print("Choose an option: ");
        int choice = scanner.nextInt();
        scanner.nextLine();

        switch (choice) {
            case 1:
                DB2024TEAM07_RestaurantManager.addRestaurant(scanner);
                break;
            case 2:
                displayAllRestaurants();
                DB2024TEAM07_RestaurantManager.updateRestaurant(scanner, new DB20
24TEAM07_RestaurantDAO());
                break;
            case 3:
                search_restaurant_options(scanner);
                break;
            case 4:
                displayAllRestaurants();
                DB2024TEAM07_RestaurantManager.deleteRestaurant(scanner);
                break;
            case 5:
                DB2024TEAM07_MenuManager.addMenu(scanner);
                break;
            case 6:
                displayAllRestaurants();
                DB2024TEAM07_MenuManager.updateMenu(scanner, new DB2024TEAM07_Men
uDAO());
                break;
            case 7:
                DB2024TEAM07_MenuManager.searchByManager(scanner);
                break;
            case 8:
                displayAllRestaurants();
                DB2024TEAM07_MenuManager.deleteMenu(scanner);
                break;
            case 9:
                DB2024TEAM07_ReviewManager.addReview(scanner);
                break;
            case 10:
                DB2024TEAM07_ReviewManager.updateReview(scanner);
                break;
            case 11:
                search_review_options(scanner);
                break;
            case 12:
                DB2024TEAM07_ReviewManager.deleteReview(scanner);
                break;
            case 13:
                userManager.addAccountByManager(scanner);
                break;
            case 14:
                userManager.displayAllUsers();
                userManager.updateAccountByManager(scanner);
                break;
            case 15:

```

```

        userManager.searchAccountByManager(scanner);
        break;
    case 16:
        userManager.displayAllUsers();
        userManager.deleteAccountByManager(scanner);
        break;
    case 17:
        return;
    default:
        System.out.println("Invalid choice. Please try again.");
    }
}

/*
}

```

- `search_restaurant_options(Scanner scanner)`

- 레스토랑 검색을 위한 서브 메뉴 옵션을 표시하는 메서드이다.
- 작동 방식: 사용자가 선택한 서브 메뉴 옵션에 따라 해당 기능을 호출한다.

```

/**
 * Presents sub-menu options for searching restaurants.
 *
 * @param scanner the input scanner
 */
public static void search_restaurant_options(Scanner scanner){
    while (true) {
        System.out.println("\n=====\\n");
        System.out.println("1. Search by Restaurant Name, Cuisine Type, Location," +
                           "Minimum Rating");
        System.out.println("2. Search by Cuisine Type");
        System.out.println("3. Exit");
        System.out.println("\n=====");
        System.out.print("Choose an option:");
        int sub_choice = scanner.nextInt();
        scanner.nextLine();

        switch (sub_choice) {
            case 1:
                DB2024TEAM07_RestaurantManager.searchRestaurant(scanner);
                break;
            case 2:
                DB2024TEAM07_RestaurantManager.searchRestaurantByCategory(scanner);
                break;
            case 3:
                return; // Exit the menu
            default:
                System.out.println("Invalid choice. Please try again.");
        }
    }
}

```

- `search_review_options(Scanner scanner)`

- 리뷰 관리를 위한 서브 메뉴 옵션을 표시하는 메서드이다.
- 작동 방식: 사용자가 선택한 서브 메뉴 옵션에 따라 해당 기능을 호출한다.

```

/**
 * Presents sub-menu options for review management.
 *
 * @param scanner the input scanner
 */
private static void search_review_options(Scanner scanner) {
    while (true) {
        System.out.println("\n=====\\n");
        System.out.println("1. Get Review Count");
        System.out.println("2. Get Reviews");
        System.out.println("3. Get User Review Count");
        System.out.println("4. Get User Reviews");
        System.out.println("5. Get Restaurant Review Count");
        System.out.println("6. Get Restaurant Reviews");
        System.out.println("7. Exit");
        System.out.println("\n=====\\n");
        System.out.print("Choose an option: ");
        int sub_choice = scanner.nextInt();
        scanner.nextLine();

        switch (sub_choice) {
            case 1:
                DB2024TEAM07_ReviewManager.getCount();
                break;
            case 2:
                DB2024TEAM07_ReviewManager.getReview(scanner);
                break;
            case 3:
                DB2024TEAM07_ReviewManager.getUserCount(scanner);
                break;
            case 4:
                DB2024TEAM07_ReviewManager.getUserReview(scanner);
                break;
            case 5:
                DB2024TEAM07_ReviewManager.getResCount(scanner);
                break;
            case 6:
                DB2024TEAM07_ReviewManager.getResReview(scanner);
                break;
            case 7:
                return;
            default:
                System.out.println("Invalid choice. Please try again.");
        }
    }
}

```

▼ package com.jdbc.model;

Classes	
Class	Description
DB2024TEAM07_Menu	DB 2024 team 07 menu.
DB2024TEAM07_Restaurant	DB 2024 team 07 restaurant.
DB2024TEAM07_Review	DB 2024 team 07 review.
DB2024TEAM07_User	DB 2024 team 07 user.
DB2024TEAM07_UserReview	DB 2024 team 07 user review.

## ▼ Menu

: DB2024\_Menu 테이블에 대한 DTO이다.

- DB2024\_User(menu\_id, menu\_name, res\_id, price)의 구조에 따라서, 클래스 내부에는 4개의 private 멤버 필드가 선언되었다.  
`(int)menu_id, (String)menu_name, (int)res_id, (int)price`
- 각 필드에 대한 getter와 setter 메서드를 제공하여 필드 값을 읽고 쓸 수 있도록 한다.

## ▼ Restaurant

: DB2024\_Restaurant 테이블에 대한 DTO이다.

- DB2024\_User(res\_name, res\_id, phone\_num, address, operating\_hours, break\_time, rating, cuisine\_type, location)의 구조에 따라서, 클래스 내부에는 9개의 private 멤버 필드가 선언되었다.  
`(String)res_name, (int)res_id, (String)phone_num, (String)address, (String)operating_hours, (String)break_time, (float)rating, (String)cuisine_type, (String)location`
- 각 필드에 대한 getter와 setter 메서드를 제공하여 필드 값을 읽고 쓸 수 있도록 한다.

## ▼ Review

: DB2024\_Review 테이블에 대한 DTO이다.

- DB2024\_Review(review\_id, user\_id, rating, review\_content)의 구조에 따라서, 클래스 내부에는 4개의 private 멤버 필드가 선언되었다.  
`(String)review_id, (String)user_id, (int)rating, (String)review_content`
- 2 종류의 Constructor(1. 인자 없음, 2. 모든 멤버 필드를 인자로 가짐)가 제공된다.
- 각 필드에 대한 getter와 setter 메서드를 제공하여 필드 값을 읽고 쓸 수 있도록 한다.

## ▼ User

: DB2024\_User 테이블에 대한 DTO이다.

- DB2024\_User(user\_id, user\_pw, name, student\_id, email, location)의 구조에 따라서, 클래스 내부에는 6개의 private 멤버 필드가 선언되었다.  
`(String)user_id, (String)user_pw, (String)name, (int)student_id, (String)email, (String)location`
- 2 종류의 Constructor(1. 인자 없음, 2. 모든 멤버 필드를 인자로 가짐)가 제공된다.
- 각 필드에 대한 getter와 setter 메서드를 제공하여 필드 값을 읽고 쓸 수 있도록 한다.

## ▼ UserReview

: DB2024\_Review 테이블과 DB2024\_OtherUser 뷰의 NATURAL JOIN 결과로 생성된 결과물에 대한 DTO이다.

- 조인 결과물(user\_id, review\_id, rating, review\_content, name, email)의 구조에 따라서, 클래스 내부에는 6개의 private 멤버 필드가 선언되었다.  
`(String)user_id, (String)review_id, (int)rating, (String)review_content, (String)name, (String)email`

- 2 종류의 Constructor(1. 인자 없음, 2. 모든 멤버 필드를 인자로 가짐)가 제공된다.
- 각 필드에 대한 getter와 setter 메서드를 제공하여 필드 값을 읽고 쓸 수 있도록 한다.

▼ package com.jdbc.database;

Classes	
Class	Description
DB2024TEAM07_Database	This class provides a single point of access to the database connection for the E-MATEASY application.
DB2024TEAM07_MenuDAO	This class provides data access object (DAO) methods for interacting with the DB2024_Menu table in a database.
DB2024TEAM07_RatingDAO	This class provides data access object (DAO) methods for interacting with the DB2024_Rating and DB2024_Review tables in a database.
DB2024TEAM07_RestaurantDAO	This class provides data access object (DAO) methods for interacting with the DB2024_Restaurant table in a database.
DB2024TEAM07_ReviewDAO	This class provides data access object (DAO) methods for interacting with the DB2024_Review and related tables in a database.
DB2024TEAM07_UserDAO	This class provides a Data Access Object (DAO) for managing users in a database.

## ▼ Database

: 데이터베이스와의 연결을 관리하는 싱글톤 클래스이다. 데이터베이스 연결을 생성한다.

```
private DB2024TEAM07_Database() {
    try {
        conn = DriverManager.getConnection(DB_URL, USER, PASS);
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
public Connection getConnection() {
    return conn;
}
```

## ▼ MenuDAO

: 메뉴를 추가, 검색, 업데이트, 삭제하는 기능을 제공하는 클래스이다.

### 1. add(DB2024TEAM07\_Menu menu)

```
public int add(DB2024TEAM07_Menu menu) {
    String Q = "INSERT INTO DB2024_Menu (menu_id, menu_name, res_id, price) VALUES (?, ?, ?, ?)";
    try {
        pStmt = conn.prepareStatement(Q);
        pStmt.setInt(1, menu.getMenu_id());
        pStmt.setString(2, menu.getMenu_name());
        pStmt.setInt(3, menu.getRes_id());
        pStmt.setInt(4, menu.getPrice());

        return pStmt.executeUpdate();

    } catch (SQLException se) {
        se.printStackTrace();
    }
    return 0;
}
```

- 새로운 메뉴를 추가하는 함수이다.

- 작동 방식:

1. `INSERT INTO DB2024_Menu (menu_id, menu_name, res_id, price) VALUES (?, ?, ?, ?)` 쿼리를 사용한다.
2. `pStmt` 객체를 사용하여 쿼리의 파라미터를 설정한다.

- 3. `pStmt.executeUpdate()` 메서드를 호출하여 쿼리를 실행하고, 결과를 반환한다.

## 2. `searchMenuByRestaurant(int res_id)`

```
public ResultSet searchMenuByRestaurant(int res_id) { // 메뉴 조회 - 식당별로 메뉴 검색
    String Q = "SELECT menu_id, menu_name, price FROM DB2024_Menu use index(DB2024_idx_Menu) WHERE res_id = ?";
    try {
        pStmt = conn.prepareStatement(Q);
        pStmt.setInt(1, res_id);
        return pStmt.executeQuery();
    } catch (SQLException se) {
        se.printStackTrace();
    }
    return null;
}
```

- 특정 식당에 존재하는 메뉴를 검색하는 함수이다.
- `res_id` 컬럼에 인덱스(`DB2024_idx_Menu`)를 사용하여 효율적으로 검색한다.
- 작동 방식:
  - `SELECT menu_id, menu_name, price FROM DB2024_Menu use index(DB2024_idx_Menu) WHERE res_id = ?` 쿼리를 사용한다.
  - `pStmt` 객체를 사용하여 쿼리의 파라미터를 설정한다.
  - `pStmt.executeQuery()` 메서드를 호출하여 쿼리를 실행하고, 결과를 반환한다.

## 3. `searchByManager(int res_id)`

```
public ResultSet searchByManager(int res_id) {
    String Q = "SELECT * FROM DB2024_Menu WHERE res_id=?";
    try {
        pStmt = conn.prepareStatement(Q);
        pStmt.setInt(1, res_id);

        return pStmt.executeQuery();
    } catch (SQLException se) {
        se.printStackTrace();
    }
    return null;
}
```

- 관리자가 식당 ID로 메뉴를 검색하는 함수이다.
- 작동 방식
  - `SELECT * FROM DB2024_Menu WHERE res_id=?` 쿼리를 사용한다.
  - `pStmt` 객체를 사용하여 쿼리의 파라미터를 설정한다.
  - `pStmt.executeQuery()` 메서드를 호출하여 쿼리를 실행하고, 결과를 반환한다.

## 3. `update(DB2024TEAM07_Menu menu, int pRes_id, int pMenu_id)`

```
public int update(DB2024TEAM07_Menu menu, int pRes_id, int pMenu_id) {

    StringBuilder queryBuilder = new StringBuilder("UPDATE DB2024_Menu SET ");
    List<Object> parameters = new ArrayList<>();

    if (menu.getMenu_id() != -1) {
```

```

        queryBuilder.append("menu_id = ?, ");
        parameters.add(menu.getMenu_id());
    }
    if (menu.getMenu_name() != null && !menu.getMenu_name().isEmpty()) {
        queryBuilder.append("menu_name = ?, ");
        parameters.add(menu.getMenu_name());
    }
    if (menu.getRes_id() != -1) {
        queryBuilder.append("res_id = ?, ");
        parameters.add(menu.getRes_id());
    }
    if (menu.getPrice() != -1) {
        queryBuilder.append("price = ?, ");
        parameters.add(menu.getPrice());
    }

    queryBuilder.setLength(queryBuilder.length() - 2);

    queryBuilder.append(" WHERE res_id = ? AND menu_id = ?");
    parameters.add(pRes_id);
    parameters.add(pMenu_id);

    String query = queryBuilder.toString();

    try {
        pStmt = conn.prepareStatement(query);

        for (int i = 0; i < parameters.size(); i++) {
            pStmt.setObject(i + 1, parameters.get(i));
        }

        return pStmt.executeUpdate();
    } catch (SQLException se) {
        se.printStackTrace();
    }
    return 0;
}

```

- 기존 메뉴를 수정하는 함수이다.
- 원하는 필드의 값만 선택적으로 수정할 수 있다.
- 작동 방식:**

1. `UPDATE DB2024_Menu SET ... WHERE res_id = ? AND menu_id = ?` 형태의 쿼리를 사용한다.
2. `pStmt` 객체를 사용하여 쿼리의 파라미터를 설정한다.
3. `pStmt.executeUpdate()` 메서드를 호출하여 쿼리를 실행하고, 결과를 반환한다.

#### 4. `delete(int res_id, int menu_id)`

```

public int delete(int res_id, int menu_id) { //메뉴 삭제 (관리자 관점)
    String Q = "DELETE FROM DB2024_Menu WHERE res_id=? AND menu_id=?";
    try {
        pStmt = conn.prepareStatement(Q);
        pStmt.setInt(1, res_id);
        pStmt.setInt(2, menu_id);

        return pStmt.executeUpdate();
    } catch (SQLException se) {

```

```

        se.printStackTrace();
    }
    return 0;
}

```

- 메뉴를 삭제하는 함수이다.
- **작동 방식:**
  1. `DELETE FROM DB2024_Menu WHERE res_id=? AND menu_id=?` 쿼리를 사용한다.
  2. `pStmt` 객체를 사용하여 쿼리의 파라미터를 설정한다.
  3. `pStmt.executeUpdate()` 메서드를 호출하여 쿼리를 실행하고, 결과를 반환한다.

##### 5. `getAllMenuByRestaurant(int res_id)`

```

public ResultSet getAllMenuByRestaurant(int res_id) {
    String query = "SELECT menu_id, menu_name FROM DB2024_Menu WHERE res_id = ?";
    try {
        PreparedStatement statement = conn.prepareStatement(query);
        statement.setInt(1, res_id);
        return statement.executeQuery();
    } catch (SQLException se) {
        se.printStackTrace();
    }
    return null;
}

```

- `res_id`를 기준으로 특정 레스토랑의 모든 메뉴를 데이터베이스에서 가져오는 메서드이다. 각 메뉴의 `menu_id` 와 `menu_name` 을 쿼리하여 결과를 `ResultSet` 으로 반환한다.
- **작동 방식:**
  1. 주어진 쿼리를 사용하여 레스토랑 ID에 해당하는 모든 메뉴를 선택한다.
  2. `PreparedStatement` 객체를 사용하여 쿼리를 실행한다. 이때 `res_id` 를 쿼리의 매개변수로 설정한다.
  3. `pStmt.executeQuery()` 메서드를 호출하여 쿼리를 실행하고, 결과를 반환한다.

## ▼ RestaurantDAO

: 식당 등록, 조회, 수정, 삭제와 같은 기능을 제공하는 클래스이다.

##### 1. `add(DB2024TEAM07_Restaurant restaurant)`

```

public int add(DB2024TEAM07_Restaurant restaurant) {
    String Q = "INSERT INTO DB2024_Restaurant (res_name, res_id, phone_num, address, operating_hours, break_time, rating, cuisine_type, location) VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?)";
    try {
        pStmt = conn.prepareStatement(Q);

        // res_name
        if (restaurant.getRes_name() != null) {
            pStmt.setString(1, restaurant.getRes_name());
        } else {
            pStmt.setNull(1, java.sql.Types.VARCHAR);
        }

        // res_id
        pStmt.setInt(2, restaurant.getRes_id()); // res_id는 반드시 있어야 한다고 가정
    }
}

```

```

    // phone_num
    if (restaurant.getPhone_num() != null) {
        pstmt.setString(3, restaurant.getPhone_num());
    } else {
        pstmt.setNull(3, java.sql.Types.VARCHAR);
    }

    // address
    if (restaurant.getAddress() != null) {
        pstmt.setString(4, restaurant.getAddress());
    } else {
        pstmt.setNull(4, java.sql.Types.VARCHAR);
    }

    // operating_hours
    if (restaurant.getOperating_hours() != null) {
        pstmt.setString(5, restaurant.getOperating_hours());
    } else {
        pstmt.setNull(5, java.sql.Types.VARCHAR);
    }

    // break_time
    if (restaurant.getBreak_time() != null) {
        pstmt.setString(6, restaurant.getBreak_time());
    } else {
        pstmt.setNull(6, java.sql.Types.VARCHAR);
    }

    // rating
    if (restaurant.getRating() != -1) {
        pstmt.setFloat(7, restaurant.getRating());
    } else {
        pstmt.setNull(7, java.sql.Types.FLOAT);
    }

    // cuisine_type
    if (restaurant.getCuisine_type() != null) {
        pstmt.setString(8, restaurant.getCuisine_type());
    } else {
        pstmt.setNull(8, java.sql.Types.VARCHAR);
    }

    // location
    if (restaurant.getLocation() != null) {
        pstmt.setString(9, restaurant.getLocation());
    } else {
        pstmt.setNull(9, java.sql.Types.VARCHAR);
    }

    return pstmt.executeUpdate();
} catch (SQLException se) {
    se.printStackTrace();
}
return 0;
}

```

- 새로운 식당을 등록하는 함수이다.
- 작동 방식:

1. `INSERT INTO DB2024_Restaurant (res_name, res_id, phone_num, address, operating_hours, break_time, rating, cuisine_type, location) VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?)` 쿼리를 사용한다.
2. `pStmt` 객체를 사용하여 각 파라미터를 설정한다.
3. `pStmt.executeUpdate()` 메서드를 호출하여 쿼리를 실행하고, 결과를 반환한다.

## 2. search(String res\_name, String cuisine\_type, String location, Float rating)

```

public List<DB2024TEAM07_Restaurant> search(String res_name, String cuisine_type, String location, Float rating) {

    StringBuilder Q = new StringBuilder("SELECT * FROM DB2024_Restaurant WHERE 1=1");
    List<Object> params = new ArrayList<>();

    if (res_name != null && !res_name.isEmpty()) {
        Q.append(" AND res_name LIKE ?");
        params.add("%" + res_name + "%");
    }
    if (cuisine_type != null && !cuisine_type.isEmpty()) {
        Q.append(" AND cuisine_type LIKE ?");
        params.add("%" + cuisine_type + "%");
    }
    if (location != null && !location.isEmpty()) {
        Q.append(" AND location LIKE ?");
        params.add("%" + location + "%");
    }
    if (rating != null) {
        Q.append(" AND rating >=?");
        params.add(rating);
    }

    List<DB2024TEAM07_Restaurant> restaurants = new ArrayList<>();
    try {
        PreparedStatement pStmt = conn.prepareStatement(Q.toString());

        for (int i = 0; i < params.size(); i++) {
            pStmt.setObject(i + 1, params.get(i));
        }
        ResultSet rs = pStmt.executeQuery();

        while (rs.next()) {
            DB2024TEAM07_Restaurant restaurant = new DB2024TEAM07_Restaurant(
                rs.getString("res_name"),
                rs.getInt("res_id"),
                rs.getString("phone_num"),
                rs.getString("address"),
                rs.getString("operating_hours"),
                rs.getString("break_time"),
                rs.getFloat("rating"),
                rs.getString("cuisine_type"),
                rs.getString("location")
            );
            restaurants.add(restaurant);
        }
    } catch (SQLException se) {
        se.printStackTrace();
    }
}

```

```
        return restaurants;
    }
```

- 복합적인 조건에 따라 식당을 검색하는 함수이다.
- 작동 방식:
  - "SELECT \* FROM DB2024\_Restaurant WHERE 1=1" 형태의 쿼리를 사용한다.
  - pStmt 객체를 사용하여 각 파라미터를 설정한다.
  - 결과를 반복하여 DB2024TEAM07\_Restaurant 객체 리스트에 추가한다.
  - 결과 리스트를 반환한다.

### 3. searchRestaurantByCategory(String cuisine\_type)

```
public ResultSet searchRestaurantByCategory(String cuisine_type) {
    String Q = "SELECT res_name, phone_num, address, operating_hours, break_time, rating, location " +
               "FROM DB2024_Restaurant USE INDEX(DB2024_idx_Restaurant) WHERE cuisine_type = ?";
    try {
        pStmt = conn.prepareStatement(Q);
        pStmt.setString(1, cuisine_type);
        return pStmt.executeQuery();
    } catch (SQLException se) {
        se.printStackTrace();
    }
    return null;
}
```

- cuisine\_type 별로 식당을 검색하는 함수이다.
- 작동 방식:
  - "SELECT res\_name, phone\_num, address, operating\_hours, break\_time, rating, location FROM DB2024\_Restaurant USE INDEX(DB2024\_idx\_Restaurant) WHERE cuisine\_type = ?" 쿼리를 사용한다.
  - pStmt 객체를 사용하여 쿼리의 파라미터를 설정한다.
  - pStmt.executeQuery() 메서드를 호출하여 쿼리를 실행하고, 결과를 반환한다.

### 4. update(DB2024TEAM07\_Restaurant restaurant, int pRes\_id)

```
public int update(DB2024TEAM07_Restaurant restaurant, int pRes_id) {
    StringBuilder queryBuilder = new StringBuilder("UPDATE DB2024_Restaurant SET ");
    List<Object> parameters = new ArrayList<>();

    if (restaurant.getRes_name() != null) {
        queryBuilder.append("res_name=?," );
        parameters.add(restaurant.getRes_name());
    }
    if (restaurant.getPhone_num() != null) {
        queryBuilder.append("phone_num=?," );
        parameters.add(restaurant.getPhone_num());
    }
    if (restaurant.getAddress() != null) {
        queryBuilder.append("address=?," );
        parameters.add(restaurant.getAddress());
    }
    if (restaurant.getOperating_hours() != null) {
```

```

        queryBuilder.append("operating_hours=?, ");
        parameters.add(restaurant.getOperating_hours());
    }
    if (restaurant.getBreak_time() != null) {
        queryBuilder.append("break_time=?, ");
        parameters.add(restaurant.getBreak_time());
    }
    if (restaurant.getRating() != -1) {
        queryBuilder.append("rating=?, ");
        parameters.add(restaurant.getRating());
    }
    if (restaurant.getCuisine_type() != null) {
        queryBuilder.append("cuisine_type=?, ");
        parameters.add(restaurant.getCuisine_type());
    }
    if (restaurant.getLocation() != null) {
        queryBuilder.append("location=?, ");
        parameters.add(restaurant.getLocation());
    }

    if (!parameters.isEmpty()) {
        queryBuilder.setLength(queryBuilder.length() - 2);
    } else {
        return 0;
    }

    queryBuilder.append(" WHERE res_id=?");
    parameters.add(pRes_id);

    String query = queryBuilder.toString();

    try {
        pStmt = conn.prepareStatement(query);

        int i = 1;
        for (Object param : parameters) {
            pStmt.setObject(i, param);
            i++;
        }
        return pStmt.executeUpdate();
    } catch (SQLException se) {
        se.printStackTrace();
    }
    return 0;
}

```

- 기존 식당의 정보를 수정하는 함수이다.
- 원하는 필드의 값만 선택적으로 수정할 수 있다.
- 작동 방식:**
  - `UPDATE DB2024_Restaurant SET ... WHERE res_id=?` 형태의 쿼리를 사용한다.
  - `pStmt` 객체를 사용하여 각 파라미터를 설정한다.
  - `pStmt.executeUpdate()` 메서드를 호출하여 쿼리를 실행하고, 결과를 반환한다.

## 5. **delete(int res\_id)**

```

public int delete(int res_id) {
    String Q = "DELETE FROM DB2024_Restaurant WHERE res_id=?";
    try {
        pStmt = conn.prepareStatement(Q);
        pStmt.setInt(1, res_id);

        return pStmt.executeUpdate();

    } catch (SQLException se) {
        se.printStackTrace();
    }
    return 0;
}

```

- 입력된 식당 ID를 기준으로 해당 식당을 삭제하는 함수이다.

- **작동 방식:**

1. `DELETE FROM DB2024_Restaurant WHERE res_id=?` 쿼리를 사용한다.
2. `pStmt` 객체를 사용하여 쿼리의 파라미터를 설정한다.
3. `pStmt.executeUpdate()` 메서드를 호출하여 쿼리를 실행하고, 결과를 반환한다.

## 6. getRandomRestaurant()

```

public static DB2024TEAM07_Restaurant getRandomRestaurant() {
    String query = "SELECT * FROM DB2024_Restaurant ORDER BY RAND() LIMIT 1";
    try {
        ResultSet rs = conn.createStatement().executeQuery(query);
        if (rs.next()) {
            return new DB2024TEAM07_Restaurant(
                rs.getString("res_name"),
                rs.getInt("res_id"),
                rs.getString("phone_num"),
                rs.getString("address"),
                rs.getString("operating_hours"),
                rs.getString("break_time"),
                rs.getFloat("rating"),
                rs.getString("cuisine_type"),
                rs.getString("location")
            );
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
    return null;
}

```

- 데이터베이스에서 무작위로 한 식당을 가져오는 함수이다.

- **작동 방식:**

1. `SELECT * FROM DB2024_Restaurant ORDER BY RAND() LIMIT 1` 쿼리를 사용한다.
2. `DB2024TEAM07_Restaurant` 객체를 생성하여 반환한다.

## 7. getAllRestaurants()

```

public List<DB2024TEAM07_Restaurant> getAllRestaurants() {
    String query = "SELECT res_id, res_name FROM DB2024_Restaurant";

```

```

List<DB2024TEAM07_Restaurant> restaurants = new ArrayList<>();

try {
    pStmt = conn.prepareStatement(query);
    ResultSet rs = pStmt.executeQuery();

    while (rs.next()) {
        DB2024TEAM07_Restaurant restaurant = new DB2024TEAM07_Restaurant(
            rs.getString("res_name"),
            rs.getInt("res_id"),
            null, // phone_num
            null, // address
            null, // operating_hours
            null, // break_time
            -1, // rating
            null, // cuisine_type
            null // location
        );
        restaurants.add(restaurant);
    }
} catch (SQLException se) {
    se.printStackTrace();
}
return restaurants;
}

```

- 모든 식당의 ID와 이름을 검색하여 반환해주는 함수이다.
- 관리자가 식당, 메뉴, 리뷰를 수정 또는 삭제를 할 때, 모든 식당의 식당 id와 식당 이름을 출력해줌으로써 원하는 식당 res\_id 를 바로 확인해서 입력할 수 있도록 도와준다.
- 작동 방식:**
  - `SELECT res_id, res_name FROM DB2024_Restaurant` 쿼리를 사용한다.
  - 결과를 반복하여 `DB2024TEAM07_Restaurant` 객체 리스트에 추가한다.

## ▼ RatingDAO

: `DB2024_Rating` 테이블에 주로 접근하는 로직을 구현한 DAO이다.

- private 필드로 (Connection)conn, (PreparedStatement)pStmt, (ResultSet) rs를 가진다.
- 인자가 없는 하나의 Constructor가 제공되며, 호출 시 conn 값이 설정된다.

### 0. 구현된 기능

- 가게에 대한 리뷰 등록: `add(int review_id, int res_id)`
- 레스토랑 평균 평점 반환: `getAvg(int res_id)`

### 1. 가게에 대한 리뷰 등록

- `add(int review_id, int res_id)`**

```

public int add(int review_id, int res_id){
    String Q = "INSERT INTO DB2024_Rating VALUES (?, ?)";
    try{
        pStmt = conn.prepareStatement(Q);
        pStmt.setInt(1, review_id);
        pStmt.setInt(2, res_id);
        return pStmt.executeUpdate();
    }catch(SQLException se){
        se.printStackTrace();
    }
}

```

```

        return -2; //error
    }
}

```

- 특정 가게에 대한 리뷰가 등록되는 경우, 그 가게와 리뷰에 대한 관계를 나타내는 릴레이션 `DB2024_Rating` 에 관계 값(튜플)을 삽입해주는 함수이다.
- 새 리뷰가 작성되는 경우(`DB2024_Review` 테이블에 새 투플이 삽입된 경우) 무조건적으로 호출되어야 한다. (트랜잭션이 요구된다.)
- **작동 방식**
  1. `INSERT INTO DB2024_Rating VALUES (?, ?)` 쿼리를 사용한다.
  2. pStmt를 통해 인자값을 `review_id` 와 `res_id` 로 설정한다.
  3. `pStmt.executeUpdate()` 로 반영된 열 값을 반환한다.
  4. SQLException 발생 시 예외에 대한 정보를 출력하고 음수값(-2)를 반환한다.

## 2. 레스토랑 평균 평점 반환

### • `getAvg(int res_id)`

```

public float getAvg(int res_id){
/*
SELECT AVG(r.rating)
FROM DB2024_Review r
    INNER JOIN DB2024_Rating ra
    USE INDEX (DB2024_idx_AvgRating) ON r.review_id = ra.review_id
WHERE ra.res_id = ?
*/
String Q = "SELECT AVG(r.rating) FROM DB2024_Review r INNER JOIN DB2024_Rating ra USE INDEX (DB2024_idx_AvgRating) ON r.review_id = ra.review_id WHERE ra.res_id = ?";
try{
    pStmt = conn.prepareStatement(Q);
    pStmt.setInt(1, res_id);
    rs = pStmt.executeQuery();
    if(rs.next()){
        return rs.getFloat(1);
    }
    return -1; //res_id is wrong
} catch(SQLException se){
    se.printStackTrace();
}
return -2; //error
}

```

- 특정 가게의 평균 평점을 반환하는 함수이다.
- 리뷰의 추가, 수정, 삭제가 발생했을 경우(`DB2024_Review` 테이블에서 데이터 삽입, 수정, 삭제가 발생한 경우) 경우 무조건적으로 호출되어야 한다. (트랜잭션이 요구된다.)
- **작동 방식**
  1. `SELECT AVG(r.rating) FROM DB2024_Review r INNER JOIN DB2024_Rating ra USE INDEX (DB2024_idx_AvgRating) ON r.review_id = ra.review_id WHERE ra.res_id = ?` 쿼리를 사용한다.
  2. pStmt를 통해 인자값을 `res_id` 로 설정한다.
  3. `pStmt.executeQuery()` 를 rs에 받아온다.
  4. rs에 값이 존재할 때(올바른 `res_id`가 입력되었을 때) 해당 식당의 평균 평점을 반환하고
  5. 값이 존재하지 않는 경우 -1을 반환한다.
  6. SQLException 발생 시 예외에 대한 정보를 출력하고 음수값(-2)를 반환한다.

## ▼ ReviewDAO

: DB2024\_Review 테이블에 주로 접근하는 로직을 구현한 DAO이다.

- private 필드로 (Connection)conn, (PreparedStatement)pStmt, (ResultSet) rs를 가진다.
- 인자가 없는 하나의 Constructor가 제공되며, 호출 시 conn 값이 설정된다.

### 0. 구현된 기능

- 리뷰 작성: add(DB2024TEAM07\_Review review)
- 리뷰 수정: update(DB2024TEAM07\_Review review)
- 리뷰 개수 반환: getCount(), getUserCount(String user\_id), getResCount(int res\_id)
- 리뷰 리스트 반환: getReview(int page), getUserReview(int page, String user\_id), getResReview(int page, String res\_id)
- 리뷰 정보 반환: getResIdForReview(Connection conn, int reviewId)
- 리뷰 삭제: delete(int review\_id)

### 1. 리뷰 작성

- **add(DB2024TEAM07\_Review review)**

```
public int add(DB2024TEAM07_Review review, int menuId, int resId) {  
    String checkUserQuery = "SELECT COUNT(*) FROM DB2024_User WHERE user_id = ?";  
    String checkMenuQuery = "SELECT COUNT(*) FROM DB2024_Menu WHERE menu_id = ? AND res_id = ?";  
    String reviewQuery = "INSERT INTO DB2024_Review (user_id, rating, review_content) VALUES (?, ?, ?)";  
    String mappingQuery = "INSERT INTO DB2024_Review_Menu_Res_Mapping (review_id, menu_id, res_id) VALUES (?, ?, ?)";  
  
    try {  
        // 먼저 user_id가 존재하는지 확인  
        try (PreparedStatement pStmtUser = conn.prepareStatement(checkUserQuery)) {  
            pStmtUser.setString(1, review.getUser_id());  
            try (ResultSet rs = pStmtUser.executeQuery()) {  
                if (rs.next() && rs.getInt(1) == 0) {  
                    // user_id가 존재하지 않음  
                    return -4;  
                }  
            }  
        }  
        // 다음으로 menuId가 resId에 해당하는 레스토랑에 존재하는지 확인  
        try (PreparedStatement pStmtCheck = conn.prepareStatement(checkMenuQuery)) {  
            pStmtCheck.setInt(1, menuId);  
            pStmtCheck.setInt(2, resId);  
            try (ResultSet rs = pStmtCheck.executeQuery()) {  
                if (rs.next() && rs.getInt(1) == 0) {  
                    // menuId가 resId에 해당하는 레스토랑에 없음  
                    return -3;  
                }  
            }  
        }  
        // 모든 검증이 통과되면 리뷰 추가 진행  
        try (PreparedStatement pStmt = conn.prepareStatement(reviewQuery, Statement.RETURN_GENERATED_KEYS)) {  
            pStmt.setString(1, review.getUser_id());  
            pStmt.setInt(2, review.getRating());  
            pStmt.setString(3, review.getReviewContent());  
            pStmt.executeUpdate();  
            try (ResultSet rs = pStmt.getGeneratedKeys()) {  
                if (rs.next()) {  
                    review.setId(rs.getInt(1));  
                }  
            }  
        }  
    }  
}
```

```

        ment.RETURN_GENERATED_KEYS)) {
            pStmt.setString(1, review.getUser_id());
            pStmt.setInt(2, review.getRating());
            pStmt.setString(3, review.getReview_content());
            pStmt.executeUpdate();

            // 생성된 리뷰 ID 가져오기
            try (ResultSet generatedKeys = pStmt.getGeneratedKeys()) {
                if (generatedKeys.next()) {
                    int reviewId = generatedKeys.getInt(1);
                    review.setReview_id(reviewId);

                    // 매핑 테이블에 데이터 추가
                    try (PreparedStatement pStmtMapping = conn.prepareStatement(
                            mappingQuery)) {
                        pStmtMapping.setInt(1, reviewId);
                        pStmtMapping.setInt(2, menuId);
                        pStmtMapping.setInt(3, resId);
                        pStmtMapping.executeUpdate();
                    }
                } else {
                    return -1; // 리뷰 ID 생성 실패
                }
            }
            return 1; // 성공
        }
    } catch (SQLException se) {
        se.printStackTrace();
        return -2; // SQL 예외 발생
    }
}

```

- 특정 가게의 특정 메뉴에 대한 리뷰를 등록하는 함수이다.

- 유저 ID 존재 여부 확인:

- 먼저 `DB2024_User` 테이블에서 해당 `user_id` 가 존재하는지 확인한다. 존재하지 않을 경우 -4를 반환하여 유저 ID 가 존재하지 않음을 알린다.

- 메뉴 ID 및 레스토랑 ID 유효성 확인:

- 다음으로 `DB2024_Menu` 테이블에서 `menu_id` 가 주어진 `res_id` 에 해당하는 레스토랑에 존재하는지 확인한다. 존재 하지 않을 경우 -3을 반환하여 메뉴 ID가 해당 레스토랑에 존재하지 않음을 알린다.

- SQL 예외 처리:

- `SQLException` 이 발생할 수 있으며, 이 경우 `se.printStackTrace()` 를 통해 스택 트레이스를 출력하고, -2를 반환하 여 SQL 예외가 발생했음을 나타낸다.

- 리뷰 ID 생성 실패 처리:

- 리뷰 ID가 생성되지 않을 경우 -1을 반환하여 리뷰 ID 생성에 실패했음을 나타낸다.

- 리뷰 추가 및 매핑 데이터 삽입:

- 모든 검증이 통과되면 실제 리뷰를 `DB2024_Review` 테이블에 추가하고, 해당 리뷰와 메뉴 및 레스토랑을 매핑하는 데이터를 `DB2024_Review_Menu_Res_Mapping` 테이블에 추가한다. 1을 반환하여 작업이 성공적으로 완료되었음을 나타낸다.

## 2. 리뷰 수정

- update(DB2024TEAM07\_Review review)**

```

public int update(DB2024TEAM07_Review review){
    String Q = "UPDATE DB2024_Review SET rating=?, review_content=? WHERE review_id=?";

```

```

try{
    pStmt = conn.prepareStatement(Q);
    pStmt.setInt(1, review.getRating());
    pStmt.setString(2, review.getReview_content());
    pStmt.setInt(3, review.getReview_id());
    return pStmt.executeUpdate();
} catch(SQLException se){
    se.printStackTrace();
}
return -2; //error
}

```

- DB2024TEAM07\_Review 타입의 객체를 인자값으로 받아 DB2024\_Review 테이블의 데이터를 수정하는 함수이다.

#### ◦ 작동 방식

- UPDATE DB2024\_Review SET rating=?, review\_content=? WHERE review\_id=? 쿼리를 사용한다.
- pStmt를 통해 인자값을 설정한다.
- SQLException 발생 시 예외에 대한 정보를 출력하고 음수값(-2)를 반환한다.

#### 3. 리뷰 개수 반환

- 페이지네이션을 위해 구현된 함수이다. DB2024\_Review 테이블의 데이터를 조건에 맞게 검색해 반환한다.
- getCount()**

```

public int getCount() {
    String Q = "SELECT COUNT(*) FROM DB2024_Review";
    try{
        pStmt = conn.prepareStatement(Q);
        rs = pStmt.executeQuery();
        if(rs.next()) {
            return rs.getInt(1);
        }
        return 0; //아무 리뷰도 없는 상태
    }
    catch(SQLException se) {
        se.printStackTrace();
    }
    return -2; //error
}

```

- getUserCount(String user\_id)**

```

public int getUserCount(String user_id) {
    String Q = "SELECT COUNT(*) FROM DB2024_Review USE INDEX (DB2024_idx_Revi
w) WHERE user_id = ?";
    try{
        pStmt = conn.prepareStatement(Q);
        pStmt.setString(1, user_id);
        rs = pStmt.executeQuery();
        if(rs.next()) {
            return rs.getInt(1);
        }
        return 0; //아무 리뷰도 없는 상태
    }
    catch(SQLException se) {
        se.printStackTrace();
    }
    return -2; //error
}

```

- **getResCount(int res\_id)**

```

public int getResCount(int res_id) {
    String Q = "SELECT COUNT(*) FROM DB2024_Rating WHERE res_id = ?";
    try{
        pStmt = conn.prepareStatement(Q);
        pStmt.setInt(1, res_id);
        rs = pStmt.executeQuery();
        if(rs.next()) {
            return rs.getInt(1);
        }
        return 0; //아무 리뷰도 없는 상태
    }
    catch(SQLException se) {
        se.printStackTrace();
    }
    return -2; //error
}

```

- **getCount() 작동 방식**

1. `SELECT COUNT(*) FROM DB2024_Review` 쿼리를 사용한다.
2. `pStmt.executeQuery()` 를 rs에 받아온다.
3. rs에 값이 존재할 때(리뷰가 존재할 때) 리뷰의 총 개수를 반환한다.
4. 값이 존재하지 않는 경우 0을 반환한다.
5. SQLException 발생 시 예외에 대한 정보를 출력하고 음수값(-2)를 반환한다.

- **getUserCount(String user\_id) 작동 방식**

1. `SELECT COUNT(*) FROM DB2024_Review USE INDEX (DB2024_idx_Review) WHERE user_id = ?` 쿼리를 사용하고, pStmt를 통해 `user_id` 를 설정한다.
2. rs에 값이 존재할 때(유저가 작성한 리뷰가 존재할 때) 해당 유저가 작성한 리뷰 개수를 반환한다는 점을 제외하고는 이하 `getCount()`과 같은 방식으로 작동한다.

- **getResCount(int res\_id) 작동 방식**

1. `SELECT COUNT(*) FROM DB2024_Rating WHERE res_id = ?` 쿼리를 사용하고, pStmt를 통해 `res_id` 를 설정한다.
2. rs에 값이 존재할 때(특정 식당에 대한 리뷰가 존재할 때) 해당 식당의 리뷰 개수를 반환한다는 점을 제외하고는 이하 `getCount()`과 같은 방식으로 작동한다.

#### 4. 리뷰 리스트 반환

- 페이지 값을 받아 `DB2024_Review` 테이블의 투플을 조회하고 리뷰를 ArrayList 형태로 반환하는 함수이다.  
페이지 당 리뷰는 10개씩 출력되며, 페이지 값은 1부터 시작하고, 가장 최신 값이 보이는 페이지가 1번 페이지라는 정책을 선택하였다.
- 현재 데이터 개수가 적은 것과 코드 상의 복잡성을 고려해 ResultSet의 absolute() API를 사용해 작성했으나. 이후 데이터값이 증가되면 데이터베이스의 생명 주기에 따라 다른 방식으로 버전 업데이트를 할 여지가 있어 보인다.
- **getReview(int page)**

```

public ArrayList<DB2024TEAM07_Review> getReview(int page){
    String Q = "SELECT * FROM DB2024_Review ORDER BY review_id DESC";
    ArrayList<DB2024TEAM07_Review> list = new ArrayList<>();
    try{
        pStmt = conn.prepareStatement(Q,
                                    ResultSet.TYPE_SCROLL_SENSITIVE,
                                    ResultSet.CONCUR_READ_ONLY);
        rs = pStmt.executeQuery();
        rs.absolute((page-1)*10);
        int i=0;
        while(rs.next() && i<10) {

```

```

        DB2024TEAM07_Review review = new DB2024TEAM07_Review(
            rs.getInt(1),
            rs.getString(2),
            rs.getInt(3),
            rs.getString(4)
        );
        list.add(review);
        i++;
    }
}
catch(SQLException se) {
    se.printStackTrace();
}
return list;
}

```

- **getUserReview(int page, String user\_id)**

```

public ArrayList<DB2024TEAM07_ResReviewVO> getUserReview(int page, String user_
id){
    String Q = "SELECT * FROM DB2024_ResReview WHERE user_id= ? ORDER BY review_
_id DESC";
    ArrayList<DB2024TEAM07_ResReviewVO> userReviews = new ArrayList<>();
    try{
        pStmt = conn.prepareStatement(Q,
            ResultSet.TYPE_SCROLL_SENSITIVE,
            ResultSet.CONCUR_READ_ONLY);
        pStmt.setString(1, user_id);
        rs = pStmt.executeQuery();
        rs.absolute((page-1)*10);
        int i=0;
        while(rs.next() && i<10) {
            DB2024TEAM07_ResReviewVO review = new DB2024TEAM07_ResReviewVO(
                rs.getInt(1),
                rs.getString(2),
                rs.getString(3),
                rs.getInt(4),
                rs.getString(5)
            );
            userReviews.add(review);
            i++;
        }
    }catch(SQLException se) {
        se.printStackTrace();
    }
    return userReviews;
}

```

- **getResReview(int page, String res\_id)**

```

public ArrayList<DB2024TEAM07_UserReview> getResReview(int page, int res_id)
{
    String Q = "SELECT * FROM DB2024_Review r NATURAL JOIN DB2024_OtherUser u WHE
RE review_id IN (SELECT review_id FROM DB2024_Rating WHERE res_id = ?) ORDER BY
review_id DESC";
    ArrayList<DB2024TEAM07_UserReview> restaurantReviews = new ArrayList<>();
    try{
        pStmt = conn.prepareStatement(Q,
            ResultSet.TYPE_SCROLL_SENSITIVE,
            ResultSet.CONCUR_READ_ONLY);

```

```

pStmt.setInt(1, res_id);
rs = pStmt.executeQuery();
rs.absolute((page-1)*10);
int i=0;
while(rs.next() && i<10) {
    DB2024TEAM07_UserReview review = new DB2024TEAM07_UserReview(
        rs.getString(1),
        rs.getInt(2),
        rs.getInt(3),
        rs.getString(4),
        rs.getString(5),
        rs.getString(6)
    );
    restaurantReviews.add(review);
    i++;
}
}
catch(SQLException se) {
    se.printStackTrace();
}
return restaurantReviews;
}

```

◦ **getReview(int page)** 작동 방식

1. `SELECT * FROM DB2024_Review ORDER BY review_id DESC` 쿼리를 사용한다.
2. `pStmt.executeQuery()` 를 rs에 받아온다.
3. `absolute((page-1)*10)` 를 통해 페이지에 해당되는 데이터 투플의 위치로 커서를 옮긴다.
4. rs에 값이 존재할 때(리뷰가 존재할 때), 최대 10개까지 DB2024TEAM07\_Review 타입 리뷰를 ArrayList에 추가한다.
5. SQLException 발생 시 예외에 대한 정보를 출력한다.
6. ArrayList를 반환한다.

◦ **getUserReview(int page, String user\_id)** 작동 방식

1. `SELECT * FROM DB2024_ResReview WHERE user_id= ? ORDER BY review_id DESC` 쿼리를 사용하고, pStmt를 통해 `user_id` 를 설정한다.
2. rs에 값이 존재할 때(유저가 작성한 리뷰가 존재할 때) 해당 유저가 작성한 DB2024TEAM07\_ResReviewVO 태입 리뷰를 ArrayList에 추가한다는 점을 제외하고는 이하 **getReview(int page)**과 같은 방식으로 작동한다.

◦ **getResReview(int page, String res\_id)**작동 방식

1. `SELECT * FROM DB2024_Review r NATURAL JOIN DB2024_OtherUser u WHERE review_id IN (SELECT review_id FROM DB2024_Rating WHERE res_id = ?) ORDER BY review_id DESC` 쿼리를 사용하고, pStmt를 통해 `res_id` 를 설정한다.
2. rs에 값이 존재할 때(특정 식당에 대한 리뷰가 존재할 때) 해당 식당의 DB2024TEAM07\_UserReview타입 리뷰를 ArrayList에 추가한다는 점을 제외하고는 이하 **getReview(int page)**과 같은 방식으로 작동한다.

5. 리뷰 정보 반환

• **getResIdForReview(Connection conn, int reviewId)**

```

private static int getResIdForReview(Connection conn, int reviewId) {
    String resIdQuery = "SELECT res_id FROM DB2024_Rating WHERE review_id = ?";
    try {
        PreparedStatement pStmt = conn.prepareStatement(resIdQuery);
        pStmt.setInt(1, reviewId);
        ResultSet rs = pStmt.executeQuery();
        if (rs.next()) {
            return rs.getInt("res_id");
        } else {
            return -1;
        }
    }
}

```

```

        }
    } catch (SQLException e) {
        e.printStackTrace();
        return -1;
    }
}

```

## 6. 리뷰 삭제

- **delete(int review\_id)**

```

public int delete(int review_id){
    String Q = "DELETE FROM DB2024_Review WHERE review_id = ?";
    try{
        pStmt = conn.prepareStatement(Q);
        pStmt.setInt(1, review_id);
        int rowsAffected = pStmt.executeUpdate();
        return rowsAffected;
    }catch(SQLException se){
        se.printStackTrace();
    }
    return -2; //error
}

```

- `DB2024_Review` 테이블의 리뷰 데이터를 삭제하는 함수이다.
- **작동 방식**
  1. `DELETE FROM DB2024_Review WHERE review_id = ?` 쿼리를 사용한다.
  2. `pStmt`을 통해 `review_id`를 설정한다.
  3. 성공적으로 모든 작업이 이루어졌을 시 쿼리문이 반영된 열 값을 반환한다.
  4. `SQLException` 발생 시 예외에 대한 정보를 출력하고 음수값(-2)을 반환한다.

## ▼ UserDAO

: `DB2024_User` 테이블에 주로 접근하는 로직을 구현한 DAO이다.

- private 필드로 (Connection)conn, (PreparedStatement)pStmt, (ResultSet) rs를 가진다.
- 인자가 없는 하나의 Constructor가 제공되며, 호출 시 conn 값이 설정된다.

## 0. 구현된 기능

- 회원가입: `add(DB2024TEAM07_User user)`
- 로그인: `sighIn(String user_id, String user_pw)`
- 회원정보 수정: `update(DB2024TEAM07_User user, String pUser_id)`
- 회원정보 확인: `getUser(String user_id)`, `getOtherUser(String user_id)`, `getAllUsers()`
- 회원탈퇴: `delete(String user_id, String user_pw)`

## 1. 회원가입

- **add(DB2024TEAM07\_User user)**

```

public int add(DB2024TEAM07_User user) {
    String Q = "INSERT INTO DB2024_User VALUES (?,?,?,?,?,?)";
    try {
        pStmt = conn.prepareStatement(Q);
        pStmt.setString(1, user.getUser_id());
        pStmt.setString(2, user.getUser_pw());
        pStmt.setString(3, user.getName());
        pStmt.setInt(4, user.getStudent_id());
    }
}

```

```

        pStmt.setString(5, user.getEmail());
        pStmt.setString(6, user.getLocation());
        return pStmt.executeUpdate();
    } catch (SQLException se) {
        se.printStackTrace();
    }
    return -2; //error
}

```

- DB2024TEAM07\_User타입의 객체를 매개변수로 받아 DB2024\_User 테이블에 유저 데이터를 삽입하는 함수이다. 회원가입 시 사용된다.

#### ◦ 작동 방식

- `INSERT INTO DB2024_User VALUES (?, ?, ?, ?, ?, ?)` 쿼리를 사용한다.
- pStmt를 통해 `user_id`, `user_pw`, `name`, `student_id`, `email`, `location`을 설정한다.
- 성공적으로 모든 작업이 이루어졌을 시 쿼리문이 반영된 열 값을 반환한다.
- SQLException 발생 시 예외에 대한 정보를 출력하고 음수값(-2)을 반환한다.

### 2. 로그인

- `signIn(String user_id, String user_pw)`

```

public int signIn(String user_id, String user_pw) {
    String Q = "SELECT user_pw FROM DB2024_User WHERE user_id = ?";
    try {
        pStmt = conn.prepareStatement(Q);
        pStmt.setString(1, user_id);
        rs = pStmt.executeQuery();
        if (rs.next()) {
            if (rs.getString(1).equals(user_pw))
                return 1; //id: 존재 / pw: 일치
            else
                return 0; //id: 존재 / pw: 불일치
        } else
            return -1; //id: 결과 없음
    } catch (SQLException se) {
        se.printStackTrace();
    }
    return -2; //error
}

```

- user\_id와 user\_pw를 매개변수로 받아 DB2024\_User 테이블의 값을 검색하고 조회하는 함수이다. 로그인 기능을 제공한다.

#### ◦ 작동 방식

- `SELECT user_pw FROM DB2024_User WHERE user_id = ?` 쿼리를 사용한다.
- pStmt를 통해 `user_id`를 설정한다.
- rs를 통해 `executeQuery()` 결과값을 받아와 전달받은 매개변수인 user\_pw와 일치하는지 확인한다.
- user\_id가 존재하고 user\_pw가 일치하는 경우 1, user\_id가 존재하고 user\_pw가 불일치하는 경우 0, user\_id가 존재하지 않는 경우 -1을 반환한다.
- SQLException 발생 시 예외에 대한 정보를 출력하고 음수값(-2)을 반환한다.

### 3. 회원정보 수정

- `update(DB2024TEAM07_User user, String pUser_id)`

```

public int update(DB2024TEAM07_User user, String pUser_id) {
    String Q = "UPDATE DB2024_User SET user_id=?, user_pw=?, name=?, student_id=?,
    email=?, location=? WHERE user_id=?";

```

```

        try {
            pStmt = conn.prepareStatement(Q);
            pStmt.setString(1, user.getUser_id());
            pStmt.setString(2, user.getUser_pw());
            pStmt.setString(3, user.getName());
            pStmt.setInt(4, user.getStudent_id());
            pStmt.setString(5, user.getEmail());
            pStmt.setString(6, user.getLocation());
            pStmt.setString(7, pUser_id);
            return pStmt.executeUpdate();
        } catch (SQLException se) {
            se.printStackTrace();
        }
        return -2; //error
    }
}

```

- DB2024TEAM07\_User 타입의 객체와 이전의 유저 아이디 값을 인자값으로 받아 DB2024\_User 테이블의 데이터를 수정하는 함수이다.

- 작동 방식**

- UPDATE DB2024\_User SET user\_id=?, user\_pw=?, name=?, student\_id=?, email=?, location=? WHERE user\_id=? 쿼리를 사용한다.
- pStmt를 통해 인자값을 설정한다.
- SQLException 발생 시 예외에 대한 정보를 출력하고 음수값(-2)를 반환한다.

#### 4. 회원정보 확인

- getUser(String user\_id)**

```

public DB2024TEAM07_User getUser(String user_id) {
    String Q = "SELECT * FROM DB2024_User WHERE user_id = ?";
    try {
        DB2024TEAM07_User user = new DB2024TEAM07_User();
        pStmt = conn.prepareStatement(Q);
        pStmt.setString(1, user_id);
        rs = pStmt.executeQuery();
        if (rs.next()) { //id: 존재
            user.setUser_id(rs.getString(1));
            user.setUser_pw(rs.getString(2));
            user.setName(rs.getString(3));
            user.setStudent_id(rs.getInt(4));
            user.setEmail(rs.getString(5));
            user.setLocation(rs.getString(6));
            return user;
        }
        //else
        //id: 불일치(결과 없음)
        //아래 리턴문에서 null이 반환됨
    } catch (SQLException se) {
        se.printStackTrace();
    }
    return null; //error, id 없음
}

```

- DB2024\_User 테이블의 데이터를 검색해 특정 user\_id를 가진 유저 정보를 확인하는 기능을 제공한다. 로그인 한 유저가 본인의 정보를 확인하는 용도이다.

- 작동 방식**

- SELECT \* FROM DB2024\_User WHERE user\_id = ? 쿼리를 사용한다.
- pStmt를 통해 인자값을 설정한다.

3. 전달받은 user\_id가 테이블에 존재하는 경우 DB2024TEAM07\_User 타입의 user를 반환하고, 그렇지 않은 경우 null을 반환한다.

4. SQLException 발생 시 예외에 대한 정보를 출력한다.

- **getOtherUser(String user\_id)**

```
public DB2024TEAM07_UserVO getOtherUser(String user_id) {  
    String Q = "SELECT * FROM DB2024_OtherUser WHERE user_id = ?";  
    try {  
        pStmt = conn.prepareStatement(Q);  
        pStmt.setString(1, user_id);  
        rs = pStmt.executeQuery();  
        if (rs.next()) { //id: 존재  
            DB2024TEAM07_UserVO user = new DB2024TEAM07_UserVO(  
                rs.getString(1),  
                rs.getString(2),  
                rs.getString(3)  
            );  
            return user;  
        }  
        //else  
        //id: 불일치(결과 없음)  
        //아래 리턴문에서 null이 반환됨  
    } catch (SQLException se) {  
        se.printStackTrace();  
    }  
    return null; //error, id 없음  
}
```

- `DB2024_OtherUser` 뷰의 데이터를 검색해 특정 user\_id를 가진 유저 정보를 확인하는 기능을 제공한다.  
다른 유저의 정보를 확인하는 용도이다.

- **작동 방식**

1. `SELECT * FROM DB2024_OtherUser WHERE user_id = ?` 쿼리를 사용한다.

2. pStmt를 통해 인자값을 설정한다.

3. 전달받은 user\_id가 테이블에 존재하는 경우 DB2024TEAM07\_UserVO 타입의 user를 반환하고, 그렇지 않은 경우 null을 반환한다.

4. SQLException 발생 시 예외에 대한 정보를 출력한다.

- **getAllUsers()**

```
public List<DB2024TEAM07_User> getAllUsers() {  
    List<DB2024TEAM07_User> users = new ArrayList<>();  
    String query = "SELECT * FROM DB2024_User";  
    try {  
        pStmt = conn.prepareStatement(query);  
        rs = pStmt.executeQuery();  
        while (rs.next()) {  
            DB2024TEAM07_User user = new DB2024TEAM07_User();  
            user.setUser_id(rs.getString("user_id"));  
            user.setUser_pw(rs.getString("user_pw"));  
            user.setName(rs.getString("name"));  
            user.setStudent_id(rs.getInt("student_id"));  
            user.setEmail(rs.getString("email"));  
            user.setLocation(rs.getString("location"));  
            users.add(user);  
        }  
    } catch (SQLException se) {  
        se.printStackTrace();  
    }
```

```

    }
    return users;
}

```

- 데이터베이스에서 모든 사용자 정보를 가져와서 `DB2024TEAM07_User` 객체의 리스트로 반환하는 메서드이다.

- 작동 방식:

- 주어진 쿼리를 사용하여 데이터베이스에서 모든 사용자 정보를 선택한다.
- `PreparedStatement` 객체를 사용하여 쿼리를 실행한다.
- 실행 결과로 얻은 `ResultSet`에서 각 행마다 반복하면서 사용자 정보를 추출한다.
- 각 사용자 정보를 `DB2024TEAM07_User` 객체로 만들어서 리스트에 추가한다.
- 모든 행을 처리한 후에는 사용자 정보가 담긴 리스트를 반환한다.
- 예외가 발생할 경우 스택 트레이스를 출력하고 빈 리스트를 반환한다.

### 5. 회원탈퇴

- `delete(String user_id, String user_pw)`

```

public int delete(String user_id, String user_pw) {
    String Q = "DELETE FROM DB2024_User WHERE user_id = ?";
    try {
        int signInRes = signIn(user_id, user_pw);
        if (signInRes == 1) { //id ok pw ok
            pStmt = conn.prepareStatement(Q);
            pStmt.setString(1, user_id);
            return pStmt.executeUpdate(); // executeUpdate()로 변경
        } else { //0: id o pw x, -1: id x, -2: error
            return signInRes;
        }
    } catch (SQLException se) {
        se.printStackTrace();
    }
    return -2; //error
}

```

- 인자값으로 `user_id`와 `user_pw`를 받아 `DB2024TEAM07_User` 테이블의 유저 데이터를 삭제하는 함수이다. 회원탈퇴 기능을 제공한다.

- 작동 방식

- `DELETE FROM DB2024_Review WHERE review_id = ?` 쿼리를 사용한다.
- 계정 확인을 위해 `signIn(user_id, user_pw)`를 실행해 값을 받아온다.
- 아이디와 비밀번호가 모두 일치하는 경우 `pStmt`을 통해 `user_id`를 설정하고 1을 반환한다.
- 그렇지 않은 경우 쿼리문은 실행되지 않고, `signInRes` 결과값(-1 or -2)을 반환한다.
- `SQLException` 발생 시 예외에 대한 정보를 출력하고 음수값(-2)을 반환한다.

▼ `package com.jdbc.view;`

Classes	
Class	Description
<code>DB2024TEAM07_ResReviewVO</code>	This class represents a Value Object (VO) for a restaurant review in the E-MATEASY application.
<code>DB2024TEAM07_UserVO</code>	This class represents a Value Object (VO) for a user in the E-MATEASY application.

### ▼ ResReviewVO

: `DB2024_ResReview` 뷰에 대한 VO이다.

- `DB2024_ResReview(review_id, user_id, res_name, rating, review_content)`의 구조에 따라서, 클래스 내부에는 6개의 private 멤버 필드가 선언되었다.

(String)review\_id, (String)user\_id, (String)res\_name, (int)rating, (String)review\_content

- 2 종류의 Constructor(1. 인자 없음, 2. 모든 멤버 필드를 인자로 가짐)가 제공된다.
- 각 필드에 대한 Getter만 제공된다.

## ▼ UserVO

: DB2024\_OtherUser 뷰에 대한 VO이다.

- DB2024\_OtherUser(user\_id, name, email)의 구조에 따라서, 클래스 내부에는 3개의 private 멤버 필드가 선언되었다.

(String)user\_id (String)name, (String)email

- 2 종류의 Constructor(1. 인자 없음, 2. 모든 멤버 필드를 인자로 가짐)가 제공된다.
- 각 필드에 대한 Getter만 제공된다.

▼ package com.manager;

Classes	Description
DB2024TEAM07_MenuManager	This class manages menu functionalities in the E-MATEASY application.
DB2024TEAM07_RestaurantManager	This class manages restaurant functionalities in the E-MATEASY application.
DB2024TEAM07_ReviewManager	This class manages review functionalities in the E-MATEASY application.
DB2024TEAM07_UserManager	This class manages user functionalities in the E-MATEASY application.

## ▼ MenuManager

: 메뉴 추가, 업데이트, 검색, 삭제하는 기능을 관리하는 클래스이다.

```
package com.manager;

import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.Scanner;

import com.jdbc.database.DB2024TEAM07_MenuDAO;
import com.jdbc.model.DB2024TEAM07_Menu;

/**
 * This class manages menu functionalities in the E-MATEASY application.
 * It provides methods for adding, updating, searching, and deleting menus
 * from the database.
 */
public class DB2024TEAM07_MenuManager {

    /**
     * An instance of the DB2024TEAM07_MenuDAO class for interacting with the menu table
     * in the database.
     */
    private static DB2024TEAM07_MenuDAO menuDAO = new DB2024TEAM07_MenuDAO();

    /*...*/
}
```

- addMenu(Scanner scanner)
  - 데이터베이스에 새로운 메뉴 항목을 추가하는 메서드이다.

- 작동 방식:

1. 사용자에게 메뉴 ID, 메뉴 이름, 레스토랑 ID 및 가격을 입력하라고 요청하고, 매개변수인 `scanner` (사용자 입력을 읽기 위한 `Scanner` 객체)로 사용자 입력을 받는다.
2. 입력된 정보를 사용하여 `DB2024TEAM07_Menu` 객체를 생성한다.
3. `menuDAO` 인스턴스를 사용하여 새로운 메뉴를 데이터베이스에 추가한다.
4. 추가가 성공하면 "Menu added successfully." 메시지를 출력하고, 실패하면 "Error adding menu." 메시지를 출력한다.

```
/**
 * Adds a new menu item to the database.
 *
 * @param scanner a Scanner object to read user input
 */
public static void addMenu(Scanner scanner) {
    System.out.print("Enter Menu ID: ");
    int menu_id = scanner.nextInt();
    scanner.nextLine();

    System.out.print("Enter Menu Name: ");
    String menu_name = scanner.nextLine();

    System.out.print("Enter Restaurant ID: ");
    int res_id = scanner.nextInt();
    scanner.nextLine();

    System.out.print("Enter Price: ");
    int price = scanner.nextInt();
    scanner.nextLine();

    DB2024TEAM07_Menu menu = new DB2024TEAM07_Menu(menu_id, menu_name, res_id, price);
    int result = menuDAO.add(menu);

    if (result > 0) {
        System.out.println("Menu added successfully.");
    } else {
        System.out.println("Error adding menu.");
    }
}
```

- `updateMenu(Scanner scanner, DB2024TEAM07_MenuDAO menuDAO)`

- 기존 메뉴 항목을 업데이트하는 메서드이다.

- 작동 방식:

1. 사용자에게 레스토랑 ID를 입력하도록 요청하고, 해당 레스토랑의 모든 메뉴 항목을 표시한다.
2. 사용자에게 메뉴 ID와 새 메뉴 이름, 새 가격을 입력하도록 요청한다. (새 메뉴 이름과 새 가격은 생략할 수 있음)
3. 입력된 정보를 사용하여 업데이트된 `DB2024TEAM07_Menu` 객체를 생성한다.
4. `menuDAO` 인스턴스를 사용하여 메뉴를 업데이트한다.
5. 업데이트가 성공하면 "Menu updated successfully." 메시지를 출력하고, 실패하면 "Error updating menu." 메시지를 출력한다.

```
/**
 * Updates an existing menu item in the database.
 *
```

```

        * @param scanner a Scanner object to read user input
        * @param menuDAO an instance of the DB2024TEAM07_MenuDAO class
        */
    public static void updateMenu(Scanner scanner, DB2024TEAM07_MenuDAO menuDA
0) {
        System.out.print("Enter Restaurant ID: ");
        int res_id = scanner.nextInt();
        scanner.nextLine();

        displayAllMenu(res_id);

        System.out.print("Enter Menu ID: ");
        int menu_id = scanner.nextInt();
        scanner.nextLine();

        System.out.print("Enter New Menu Name (Press enter to skip): ");
        String newMenuName = scanner.nextLine();

        System.out.print("Enter New Price (Press enter to skip): ");
        String priceInput = scanner.nextLine();
        int newPrice = -1;
        if (!priceInput.isEmpty()) {
            newPrice = Integer.parseInt(priceInput);
        }

        DB2024TEAM07_Menu updatedMenu = new DB2024TEAM07_Menu(
            menu_id,
            newMenuName.isEmpty() ? null : newMenuName,
            res_id,
            newPrice
        );

        int result = menuDAO.update(updatedMenu, res_id, menu_id);

        if (result > 0) {
            System.out.println("Menu updated successfully.");
        } else {
            System.out.println("Error updating menu.");
        }
    }
}

```

- `searchMenuByRestaurant(Scanner scanner)`

- 특정 레스토랑 ID로 메뉴를 검색하는 메서드이다.
- 작동 방식:
  1. 사용자에게 레스토랑 ID를 입력하도록 요청한다.
  2. `menuDAO` 인스턴스를 사용하여 해당 레스토랑의 메뉴를 검색한다.
  3. 검색된 결과((메뉴 ID, 메뉴 이름 및 가격)를 출력한다.
  4. 결과가 없으면 "No data found." 메시지를 출력한다.

```

    /**
     * Searches for menus by a specific restaurant ID.
     *
     * @param scanner a Scanner object to read user input
     */
    public static void searchMenuByRestaurant(Scanner scanner) {

```

```

        System.out.print("Enter Restaurant ID: ");
        int restaurantId = scanner.nextInt();
        scanner.nextLine();

        ResultSet rs = menuDAO.searchMenuByRestaurant(restaurantId);
        try {
            if (rs != null && rs.next()) {
                System.out.println("Menu ID\tMenu Name\tPrice");
                System.out.println("-----");
                do {
                    int menuId = rs.getInt("menu_id");
                    String menuName = rs.getString("menu_name") != null ? rs.getString("menu_name") : "정보 없음";
                    String price = rs.getString("price") != null ? String.valueOf(rs.getInt("price")) : "정보 없음";

                    System.out.printf("%-10d %-20s %-10s%n", menuId, menuName, price);
                } while (rs.next());
            } else {
                System.out.println("No data found.");
            }
        } catch (SQLException se) {
            se.printStackTrace();
        }
    }
}

```

- `searchByManager(Scanner scanner)`
  - 특정 레스토랑의 메뉴를 매니저가 검색할 때 사용되는 메서드이다.
  - 작동 방식:
    1. 사용자에게 레스토랑 ID를 입력하도록 요청한다.
    2. `menuDAO` 인스턴스를 사용하여 해당 레스토랑의 메뉴를 검색한다.
    3. 검색된 결과(메뉴 ID, 레스토랑 ID, 메뉴 이름 및 가격)를 출력한다.
    4. 결과가 없으면 "No data found." 메시지를 출력한다.

```

/**
 * Searches for menus by a manager for a specific restaurant.
 *
 * @param scanner a Scanner object to read user input
 */
public static void searchByManager(Scanner scanner) {
    System.out.print("Enter Restaurant ID: ");
    int resId = scanner.nextInt();

    ResultSet rs = menuDAO.searchByManager(resId);
    try {
        if (rs != null && rs.next()) {
            System.out.println("Menu ID\tRestaurant ID\tMenu Name\tPrice");
            System.out.println("-----");
            do {
                int menuId = rs.getInt("menu_id");

```

```

        int restaurantId = rs.getInt("res_id");
        String menuName = rs.getString("menu_name") != null ? rs.get
tString("menu_name") : "정보 없음";
        String price = rs.getString("price") != null ? String.value
Of(rs.getInt("price")) : "정보 없음";

        System.out.printf("%-8d %-13d %-20s %-10s%n", menuId, resta
urantId, menuName, price);
    } while (rs.next());
} else {
    System.out.println("No data found.");
}
} catch (SQLException se) {
    se.printStackTrace();
}

}

```

- `deleteMenu(Scanner scanner)`

- 데이터베이스에서 메뉴 항목을 삭제하는 메서드이다.
- 작동 방식:
  1. 사용자에게 레스토랑 ID를 입력하도록 요청한다.
  2. 해당 레스토랑의 모든 메뉴 항목을 표시한다.
  3. 사용자에게 삭제할 메뉴 ID를 입력하도록 요청한다.
  4. `menuDAO` 인스턴스를 사용하여 메뉴를 삭제한다.
- 5. 삭제가 성공하면 "Menu deleted successfully." 메시지를 출력하고, 실패하면 "Error deleting menu." 메시지를 출력한다.

```

/**
 * Deletes a menu item from the database.
 *
 * @param scanner a Scanner object to read
 */
public static void deleteMenu(Scanner scanner) {
    System.out.print("Enter Restaurant ID: ");
    int res_id = scanner.nextInt();
    scanner.nextLine();

    displayAllMenu(res_id);

    System.out.print("Enter Menu ID: ");
    int menu_id = scanner.nextInt();
    scanner.nextLine();

    int result = menuDAO.delete(res_id, menu_id);

    if (result > 0) {
        System.out.println("Menu deleted successfully.");
    } else {
        System.out.println("Error deleting menu.");
    }
}

```

- `displayAllMenu(int res_id)`
  - 특정 레스토랑의 모든 메뉴 항목을 표시하는 메서드이다.
  - 작동 방식:
    1. `menuDAO` 인스턴스를 사용하여 해당 레스토랑의 모든 메뉴를 검색한다.
    2. 메뉴 ID와 메뉴 이름을 출력한다.
    3. 결과가 없으면 "No data found." 메시지를 출력한다.

```

    /**
     * Displays all menu items for a specific restaurant.
     *
     * @param res_id the ID of the restaurant
     */
    public static void displayAllMenu(int res_id) {
        ResultSet resultSet = menuDAO.getAllMenuByRestaurant(res_id);

        // Print header
        System.out.println("-----");
        System.out.println("Menu ID\t\tMenu Name");
        System.out.println("-----");

        try {
            while (resultSet != null && resultSet.next()) {
                System.out.printf("%-8d\t%-30s%n", resultSet.getInt("menu_id"),
resultSet.getString("menu_name"));
            }
        } catch (SQLException se) {
            se.printStackTrace();
        }

        // Footer
        System.out.println("-----");
    }
}

```

## ▼ RestaurantManager

: 레스토랑 항목을 추가, 업데이트, 검색 및 삭제하는 기능을 관리하는 클래스이다.

```

package com.manager;

import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.*;
import com.jdbc.database.DB2024TEAM07_RestaurantDAO;
import com.jdbc.model.DB2024TEAM07_Restaurant;

/**
 * This class manages restaurant functionalities in the E-MATEASY application.
 * It provides methods for adding, updating, searching, and deleting restaurants
 * from the database.
 */

public class DB2024TEAM07_RestaurantManager {
    private static DB2024TEAM07_RestaurantDAO restaurantDAO = new DB2024TEAM07_Resta
urantDAO();

    /*...*/
}

```

```
}
```

- addRestaurant(Scanner scanner)
  - 데이터베이스에 새로운 레스토랑을 추가하는 메서드이다.
  - 작동 방식:
    1. 사용자에게 레스토랑 이름, 레스토랑 ID, 전화번호, 주소, 영업 시간, 휴식 시간, 평점, 요리 유형, 위치를 입력하도록 요청한다.
    2. 입력된 정보를 사용하여 `DB2024TEAM07_Restaurant` 객체를 생성한다.
    3. `restaurantDAO` 인스턴스를 사용하여 새로운 레스토랑을 데이터베이스에 추가한다.
    4. 추가가 성공하면 "Restaurant added successfully." 메시지를 출력하고, 실패하면 "Error adding restaurant." 메시지를 출력한다.

```
/*
 * Adds a new restaurant to the database.
 *
 * @param scanner a Scanner object to read user input
 */
public static void addRestaurant(Scanner scanner) {
    System.out.print("Enter Restaurant Name: ");
    String res_name = scanner.nextLine();

    System.out.print("Enter Restaurant ID: ");
    int res_id = scanner.nextInt();
    scanner.nextLine();

    System.out.print("Enter Phone Number: ");
    String phone_num = scanner.nextLine();

    System.out.print("Enter Address: ");
    String address = scanner.nextLine();

    System.out.print("Enter Operating Hours: ");
    String operating_hours = scanner.nextLine();

    System.out.print("Enter Break Time: ");
    String break_time = scanner.nextLine();

    System.out.print("Enter Rating (choose from 1 to 5): ");
    int rating = scanner.nextInt();
    scanner.nextLine();

    System.out.print("Enter Cuisine Type: ");
    String cuisine_type = scanner.nextLine();

    System.out.print("Enter Location: ");
    String location = scanner.nextLine();

    DB2024TEAM07_Restaurant restaurant = new DB2024TEAM07_Restaurant(
        res_name, res_id, phone_num,
        address, operating_hours, break_time,
        rating, cuisine_type, location);
    int result = restaurantDAO.add(restaurant);

    if (result > 0) {
```

```

        System.out.println("Restaurant added successfully.");
    } else {
        System.out.println("Error adding restaurant.");
    }
}

```

- `updateRestaurant(Scanner scanner, DB2024TEAM07_RestaurantDAO restaurantDAO)`
  - 기존 레스토랑 정보를 업데이트하는 메서드이다.
  - 작동 방식:
    1. 사용자에게 레스토랑 ID를 입력하도록 요청한다.
    2. 새 레스토랑 이름, 새 전화번호, 새 주소, 새 영업 시간, 새 휴식 시간, 새 평점, 새 요리 유형, 새 위치를 입력하도록 요청한다 (이 값들은 생략할 수 있음).
    3. 입력된 정보를 사용하여 업데이트된 `DB2024TEAM07_Restaurant` 객체를 생성한다.
    4. `restaurantDAO` 인스턴스를 사용하여 레스토랑 정보를 업데이트한다.
  - 5. 업데이트가 성공하면 "Restaurant updated successfully." 메시지를 출력하고, 실패하면 "Failed to update restaurant." 메시지를 출력한다.

```

/**
 * Updates an existing restaurant in the database.
 *
 * @param scanner a Scanner object to read user input
 * @param restaurantDAO an instance of the DB2024TEAM07_RestaurantDAO class
 */
public static void updateRestaurant(Scanner scanner, DB2024TEAM07_RestaurantDAO restaurantDAO) {
    System.out.print("Enter Restaurant ID: ");
    int res_id = scanner.nextInt();
    scanner.nextLine();

    System.out.print("Enter New Restaurant Name (Press enter to skip): ");
    String res_name = scanner.nextLine();

    System.out.print("Enter New Phone Number (Press enter to skip): ");
    String phone_num = scanner.nextLine();

    System.out.print("Enter New Address (Press enter to skip): ");
    String address = scanner.nextLine();

    System.out.print("Enter New Operating Hours (Press enter to skip): ");
    String operating_hours = scanner.nextLine();

    System.out.print("Enter New Break Time (Press enter to skip): ");
    String break_time = scanner.nextLine();

    System.out.print("Enter New Rating (Press enter to skip): ");
    String ratingInput = scanner.nextLine();
    int rating = -1;
    if (!ratingInput.isEmpty()) {
        rating = Integer.parseInt(ratingInput);
    }

    System.out.print("Enter New Cuisine Type (Press enter to skip): ");
    String cuisine_type = scanner.nextLine();

    System.out.print("Enter New Location (Press enter to skip): ");

```

```

        String location = scanner.nextLine();

        DB2024TEAM07_Restaurant updatedRestaurant = new DB2024TEAM07_Restaurant
(
    res_name.isEmpty() ? null : res_name,
    res_id,
    phone_num.isEmpty() ? null : phone_num,
    address.isEmpty() ? null : address,
    operating_hours.isEmpty() ? null : operating_hours,
    break_time.isEmpty() ? null : break_time,
    rating,
    cuisine_type.isEmpty() ? null : cuisine_type,
    location.isEmpty() ? null : location
);

        int updateRestaurantResult = restaurantDAO.update(updatedRestaurant, re
s_id);
        if (updateRestaurantResult > 0) {
            System.out.println("Restaurant updated successfully.");
        } else {
            System.out.println("Failed to update restaurant.");
        }
}

```

- `searchRestaurant(Scanner scanner)`
  - 다양한 기준에 따라 레스토랑을 검색하는 메서드이다.
  - 작동 방식:
    1. 사용자에게 레스토랑 이름, 요리 유형, 위치, 최소 평점을 입력하도록 요청한다 (이 값들은 생략할 수 있음).
    2. 입력된 기준에 따라 `restaurantDAO` 인스턴스를 사용하여 레스토랑을 검색한다.
    3. 검색된 결과(레스토랑 이름, ID, 전화번호, 주소, 영업 시간, 휴식 시간, 평점, 요리 유형, 위치)를 출력한다.
    4. 결과가 없으면 "No restaurants found matching the criteria." 메시지를 출력한다.

```

/*
 * Search by Restaurant Name, Cuisine Type, Location, Minimum Rating */
/**
 * Searches for restaurants based on various criteria.
 *
 * @param scanner a Scanner object to read user input
 */
public static void searchRestaurant(Scanner scanner) {
    System.out.print("Enter Restaurant Name (or press Enter to skip): ");
    String restaurantName = scanner.nextLine();
    if (restaurantName.trim().isEmpty()) restaurantName = null;

    System.out.print("Enter Cuisine Type (or press Enter to skip): ");
    String cuisineType = scanner.nextLine();
    if (cuisineType.trim().isEmpty()) cuisineType = null;

    System.out.print("Enter Location (or press Enter to skip): ");
    String location = scanner.nextLine();
    if (location.trim().isEmpty()) location = null;

    System.out.print("Enter Minimum Rating (or press Enter to skip): ");
    String ratingInput = scanner.nextLine();
    Float rating = null;
    if (!ratingInput.trim().isEmpty()) {

```

```

        try {
            rating = Float.parseFloat(ratingInput);
        } catch (NumberFormatException e) {
            System.out.println("Invalid rating input. Please enter a valid
number or press Enter to skip.");
            return;
        }
    }

    List<DB2024TEAM07_Restaurant> restaurants = restaurantDAO.search(restau
rantName, cuisineType, location, rating);

    if (restaurants.isEmpty()) {
        System.out.println("No restaurants found matching the criteria.");
    } else {
        System.out.println("Restaurants found:");
        System.out.printf("%-20s \t%-10s \t%-15s \t%-30s \t%-20s \t%-20s
\t%-10s \t%-20s \t%-20s%n",
                            "Name", "ID", "Phone Number", "Address", "Operating Hours",
                            "Break Time", "Rating", "Cuisine Type", "Location");
        System.out.println("-----");
        -----
        -----
        -----");
        for (DB2024TEAM07_Restaurant restaurant : restaurants) {
            String resName = restaurant.getRes_name() != null ? restaurant.
getRes_name() : "정보 없음";
            int resId = restaurant.getRes_id(); // ID는 null이 아니라고 가정
            String phoneNum = restaurant.getPhone_num() != null ? restauran
t.getPhone_num() : "정보 없음";
            String address = restaurant.getAddress() != null ? restaurant.g
etAddress() : "정보 없음";
            String operatingHours = restaurant.getOperating_hours() != null
? restaurant.getOperating_hours() : "정보 없음";
            String breakTime = restaurant.getBreak_time() != null ? restaur
ant.getBreak_time() : "정보 없음";
            float _rating = restaurant.getRating(); // Rating은 기본값으로 가정
            String _cuisineType = restaurant.getCuisine_type() != null ? re
staurant.getCuisine_type() : "정보 없음";
            String _location = restaurant.getLocation() != null ? restauran
t.getLocation() : "정보 없음";

            System.out.printf("%-20s \t%-10d \t%-15s \t%-30s \t%-20s \t%-20
s \t%-10.1f \t%-20s \t%-20s%n",
                            resName, resId, phoneNum, address, operatingHours, brea
kTime, _rating, _cuisineType, _location);
        }
    }
}

```

- searchRestaurantByCategory(Scanner scanner)
  - 요리 유형별로 레스토랑을 검색하는 메서드이다.
  - 작동 방식:
    1. 사용자에게 요리 유형을 입력하도록 요청한다.
    2. `restaurantDAO` 인스턴스를 사용하여 해당 요리 유형의 레스토랑을 검색한다.
    3. 검색된 결과(레스토랑 이름, 전화번호, 주소, 영업 시간, 휴식 시간, 평점, 위치)를 출력한다.

4. 결과가 없으면 "No restaurants found for the given cuisine type." 메시지를 출력한다.

```

/* Search by Cuisine Type */
/**
 * Searches for restaurants by cuisine type.
 *
 * @param scanner a Scanner object to read user input
 */
public static void searchRestaurantByCategory(Scanner scanner) {
    System.out.print("Enter Cuisine Type: ");
    String cuisineType = scanner.nextLine();

    try (ResultSet rs = restaurantDAO.searchRestaurantByCategory(cuisineType)) {
        if (rs != null && rs.next()) {
            System.out.printf("%-25s %-20s %-40s %-20s %-15s %-10s %-20s%n",
                "Restaurant Name", "Phone Number", "Address", "Operating Hours",
                "Break Time", "Rating", "Location");
            System.out.println("-----");
            do {
                String resName = rs.getString("res_name") != null ? rs.getString("res_name") : "정보 없음";
                String phoneNum = rs.getString("phone_num") != null ? rs.getString("phone_num") : "정보 없음";
                String address = rs.getString("address") != null ? rs.getString("address") : "정보 없음";
                String operatingHours = rs.getString("operating_hours") != null ? rs.getString("operating_hours") : "정보 없음";
                String breakTime = rs.getString("break_time") != null ? rs.getString("break_time") : "정보 없음";
                String rating = rs.getString("rating") != null ? String.valueOf(rs.getFloat("rating")) : "정보 없음";
                String location = rs.getString("location") != null ? rs.getString("location") : "정보 없음";

                System.out.printf("%-25s %-20s %-40s %-20s %-15s %-10s %-20s%n",
                    resName, phoneNum, address, operatingHours, breakTime, rating, location);
            } while (rs.next());
        } else {
            System.out.println("No restaurants found for the given cuisine type.");
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

```

- deleteRestaurant(Scanner scanner)
  - 데이터베이스에서 레스토랑을 삭제하는 메서드이다.
  - 작동 방식:
    1. 사용자에게 레스토랑 ID를 입력하도록 요청한다.

2. `restaurantDAO` 인스턴스를 사용하여 해당 레스토랑을 삭제한다.
3. 삭제가 성공하면 "Restaurant deleted successfully." 메시지를 출력하고, 실패하면 "Failed to delete restaurant." 메시지를 출력한다.

```
/**
 * Deletes a restaurant from the database.
 *
 * @param scanner a Scanner object to read user input
 */
public static void deleteRestaurant(Scanner scanner) {
    System.out.print("Enter Restaurant ID: ");
    int res_id = scanner.nextInt();
    scanner.nextLine();

    int deleteRestaurantResult = restaurantDAO.delete(res_id);
    if (deleteRestaurantResult > 0) {
        System.out.println("Restaurant deleted successfully.");
    } else {
        System.out.println("Failed to delete restaurant.");
    }
}
```

- `displayAllRestaurant()`
    - 데이터베이스에 있는 모든 레스토랑 목록을 표시하는 메서드이다.
    - 작동 방식:
      1. `restaurantDAO` 인스턴스를 사용하여 모든 레스토랑을 검색한다.
      2. 레스토랑 ID와 이름을 출력한다.
- ```
/**
 * Displays a list of all restaurants in the database.
 */
public static void displayAllRestaurants() {
    List<DB2024TEAM07_Restaurant> restaurants = restaurantDAO.getAllRestaurants();

    // Print header
    System.out.println("-----");
    System.out.printf("%-15s %-30s%n", "Restaurant ID", "Restaurant Name");
    System.out.println("-----");

    // Print each restaurant (ID and Name)
    for (DB2024TEAM07_Restaurant restaurant : restaurants) {
        System.out.printf("%-15d %-30s%n", restaurant.getRes_id(), restaurant.getRes_name());
    }

    // Footer
    System.out.println("-----");
}
```

- `displayRandomRestaurant()`

- 데이터베이스에서 무작위로 선택된 레스토랑을 표시하는 메서드이다.
- 작동 방식:
  - `restaurantDAO` 인스턴스를 사용하여 무작위로 선택된 레스토랑을 검색한다.
  - 선택된 레스토랑의 모든 정보를 출력한다.
  - 선택된 레스토랑이 없으면 "Failed to get restaurant information." 메시지를 출력한다.

```

/**
 * Displays a randomly selected restaurant from the database.
 */
public static void displayRandomRestaurant() {
    System.out.println("Randomly recommended restaurants for you!");
    DB2024TEAM07_Restaurant randomRestaurant = DB2024TEAM07_RestaurantDAO.getRandomRestaurant();

    if (randomRestaurant != null) {
        System.out.println("-----");
        System.out.println("-----");
        System.out.printf("%-25s%-10s%-20s%-30s%-25s%-10s%-20s%-10s%n",
                           "Name", "ID", "Phone", "Address", "Operating hour", "Break time", "Rating",
                           "Cuisine Type", "Location");
        System.out.println("-----");
        System.out.println("-----");
        System.out.printf("%-25s%-10d%-20s%-30s%-25s%-10.1f%-20s%-10s%n",
                           randomRestaurant.getRes_name(),
                           randomRestaurant.getRes_id(),
                           randomRestaurant.getPhone_num() != null ? randomRestaurant.getPhone_num() : "정보 없음",
                           randomRestaurant.getAddress() != null ? randomRestaurant.getAddress() : "정보 없음",
                           randomRestaurant.getOperating_hours() != null ? randomRestaurant.getOperating_hours() : "정보 없음",
                           randomRestaurant.getBreak_time() != null ? randomRestaurant.getBreak_time() : "정보 없음",
                           randomRestaurant.getRating(),
                           randomRestaurant.getCuisine_type() != null ? randomRestaurant.getCuisine_type() : "정보 없음",
                           randomRestaurant.getLocation() != null ? randomRestaurant.getLocation() : "정보 없음");
        System.out.println("-----");
        System.out.println("-----");
    } else {
        System.out.println("Failed to get restaurant information.");
    }
}

```

## ▼ ReviewManager

: 리뷰 추가, 업데이트, 전체/사용자/식당 별 검색, 삭제하는 기능을 관리하는 클래스이다.

```

package com.manager;

import com.jdbc.database.DB2024TEAM07_Database;
import com.jdbc.database.DB2024TEAM07_ReviewDAO;

```

```

import com.jdbc.database.DB2024TEAM07_RatingDAO;
import com.jdbc.model.DB2024TEAM07_Review;
import com.jdbc.model.DB2024TEAM07_UserReview;
import com.jdbc.view.DB2024TEAM07_ResReviewVO;

import java.sql.*;import java.util.*;

/**
 * This class manages review functionalities in the E-MATEASY application.
 * It provides methods for adding, updating, deleting, and retrieving reviews
 * from the database.
 */

public class DB2024TEAM07_ReviewManager {
    private static DB2024TEAM07_ReviewDAO reviewDAO = new DB2024TEAM07_ReviewDAO();
    private static DB2024TEAM07_RatingDAO ratingDAO = new DB2024TEAM07_RatingDAO();
    private static int page;

    /*...*/
}

```

- `addReview(Scanner scanner)`

- 입력한 식당의 메뉴에 대한 새로운 리뷰와 평점을 추가하는 함수이다.
- 작동 방식:
  1. user ID, restaurant ID, menu ID, rating, review comment에 대한 사용자 입력을 받는다.
  2. 트랜잭션을 시작한다.
  3. `reviewDAO.add()` 와 `ratingDAO.add()` 를 호출해 리뷰와 평점을 추가한다.
  4. `ratingDAO.getAvg(resId)` 로 새로운 평균 평점을 계산하고 `UPDATE DB2024_Restaurant SET rating = ? WHERE res_id = ?` 쿼리를 사용해 식당의 평점으로 업데이트한다.
  5. 모든 작업이 성공하면 트랜잭션을 커밋하고, 그렇지 않으면 롤백한다.

```

/**      * Adds a new review to the database for a specific restaurant and me
nu item.      * @param scanner a Scanner object to read user input
*/
public static void addReview(Scanner scanner) {
    System.out.print("Enter User ID: ");           String userId = scanner.nextLine();
    System.out.print("Enter Restaurant ID: ");       int resId = scanner.nextInt();
    System.out.print("Enter Menu ID: ");            int menuId = scanner.nextInt();
    System.out.print("Enter Rating (1-5): ");        int rating = scanner.nextInt();
    System.out.print("Enter review comment: ");       scanner.nextLine();
() // 버퍼 비우기           String reviewContent = scanner.nextLine();
    Connection conn = null;           try {           conn = DB2024TEAM07_
Database.getInstance().getConnecion();           conn.setAutoCommit(false);
    }
    DB2024TEAM07_Review review = new DB2024TEAM07_Review(0, userId,
rating, reviewContent);           int result = reviewDAO.add(review, menuI
d, resId);
    switch (result) {           case 1:           System.out.println("Review added successfully.");
    int reviewId = review.getReview_id();
    int ratingResult = ratingDAO.add(reviewId, resId);
}

```

```

        if (ratingResult > 0) {                               double ne
wAvgRating = ratingDAO.getAvg(resId);                      if (newAvgRatin
g >= 0) {   String updateRatingQuery = "UPDATE DB20
24_Restaurant SET rating = ? WHERE res_id = ?";
try (PreparedStatement pStmt = conn.prepareStatement(updateRatingQuery)) {
    pStmt.setDouble(1, newAvgRating);
    pStmt.setInt(2, resId);
    pStmt.executeUpdate();
    System.out.println("Restaurant rating updated successfully.");
} else {   System.out.println("Failed to retrieve new average rating.");
    nn.rollback();   } else {
System.out.println("Failed to add rating.");                  conn.rollback();
    lback();  break;           case
-3:   System.out.println("Review not added: Menu ID does no
t exist for the given restaurant.");
    conn.rollback();                                     break;           case -4:
System.out.println("Review not added: User ID does not exist.");
    conn.rollback();                                     break;           default:
System.out.println("Failed to add review due to an unknown error.");
    conn.rollback();                                     break;           }
e.printStackTrace();                                 try {           } catch (SQLException e) {
conn.rollback();   if (conn != null) {
ex.printStackTrace();                                     } catch (SQLException ex) {
    try {   } finally {           if (conn != null)
        conn.setAutoCommit(true);
    e.printStackTrace();                                }
} catch (SQLException e) {                                }
} } } }

```

- `updateReview(Scanner scanner)`
  - 이미 존재하는 리뷰와 평점을 수정하는 함수이다.
  - 작동 방식:
    1. 수정할 리뷰의 ID와 새 평점, 리뷰 내용을 입력받는다.
    2. 트랜잭션을 시작한다.
    3. `reviewDAO.update()` 를 호출해 리뷰를 업데이트한다.
    4. `getResIdForReview(conn, reviewId)` 로 식당 ID를 검색해 `ratingDAO.getAvg(resId)` 로 새로운 평균 평점을 계산하고 `UPDATE DB2024_Restaurant SET rating = ? WHERE res_id = ?` 쿼리를 사용해 식당의 평점으로 업데이트한다.
    5. 모든 작업이 성공하면 트랜잭션을 커밋하고, 그렇지 않으면 롤백한다.

```

/**      * Updates an existing review in the database.      *      * @param sca
nner a Scanner object to read user input      */     public static void update
Review(Scanner scanner) {         System.out.print("Enter review ID to updat
e: ");
int reviewId = scanner.nextInt();
System.out.print("Enter rating (1-5) to update: ");
int rating = scanner.nextInt();
System.out.print("Enter review content to update: ");
scanner.nextLine();
String reviewContent = scanner.nextLine();
Connection conn = null;
try {
    conn = DB2024TEAM07_
Database.getInstance().getConnection();
    conn.setAutoCommit(false);
DB2024TEAM07_Review review = new DB2024TEAM07_Review(reviewId, n
ull, rating, reviewContent);
    int result = reviewDAO.update(revie
w);
if (result > 0) {                               System.out.println("Review upda
ted successfully.");
    int resId = getResIdForReview(conn, reviewId);
}
}

```

```

        double newAvgRating = ratingDAO.getAvg(resId);
        String updateRatingQuery = "UPDATE DB2024_Restaurant SET rating = ? WHERE re
        s_id = ?"; try { PreparedStatement pStmt =
        conn.prepareStatement(updateRatingQuery); pStmt.setDouble
        (1, newAvgRating); pStmt.setInt(2, resId); System.out.println("Restaurant rat
        ing updated successfully."); } catch (SQLException e) {
        e.printStackTrace(); }
        conn.commit();
    } else { System.out.println("Failed to update rev
        iew."); } } catch (SQLException e) { if (conn !
        = null) { try { conn.rollback(); } catch (SQLException ex) { ex.printStackTrace();
        } } e.printStackTrace(); } finally { if (conn != null) { try { conn.setAutoComm
        it(true); } catch (SQLException e) { e.printStackTrace(); } } } }
    
```

- `getCount()`

- 전체 리뷰의 수를 검색하는 함수이다.
- 작동 방식:

1. `reviewDAO.getCount()` 를 호출해 전체 리뷰의 수를 반환한다.

```

/**      * Retrieves the total number of reviews in the database.      */
public static void getCount() { int count = reviewDAO.getCount();
System.out.println("Total number of reviews: " + count); }
    
```

- `getReview(Scanner scanner)`

- 페이지를 검색해 리뷰를 보여주는 함수이다.
- 작동 방식:

1. 페이지 번호를 입력받는다.

2. `reviewDAO.getReview(page)` 를 호출해 입력받은 페이지에 있는 리뷰를 보여준다.

```

/**      * Retrieves a page of reviews from the database.      *      * @param
scanner a Scanner object to read user input      */ public static void get
Review(Scanner scanner) { System.out.print("Enter page number: ");
int page = scanner.nextInt();
ArrayList<DB2024TEAM07_Review> reviews = reviewDAO.getReview(page);
System.out.printf("%-10s%-18s%-10s%100s%n", "Review ID", "User ID",
"Rating", "Review Content"); System.out.println("-----");
-----;
-----; for (DB2024TEAM07_Review review : reviews) {
System.out.printf("%-10d%-20s%-9d%s%n",
review.getReview_
id(), review.getUser_id(), review.getR
ating(), review.getReview_content()); }
    
```

- `getUserCount(Scanner scanner)`

- 입력받은 사용자에 대한 총 리뷰 수를 검색하는 함수이다.
- 작동 방식:

1. user ID를 입력받는다.

2. `reviewDAO.getUserCount(userId)` 를 호출해 특정 사용자가 작성한 총 리뷰 수를 보여준다.

```
/**      * Retrieves the total number of reviews for a specific user.      *
 * @param scanner a Scanner object to read user input      */    public static
void getUserCount(Scanner scanner) {      System.out.print("Enter user ID:
");      String userId = scanner.nextLine();      int count = reviewDAO.getUserCount(userId);      if (count == -1)
{          System.out.println("User ID not found.");      } else if (cou
nt == -2) {          System.out.println("An error occurred while retrievin
g data.");      } else {          System.out.println("Total number of re
views for user " + userId + ": " + count);      } }
```

- `getUserReview(Scanner scanner)`

- 입력받은 사용자에 대한 리뷰 페이지를 검색해 리뷰를 보여주는 함수이다.

- 작동 방식:

1. user ID를 입력받는다.

2. `reviewDAO.getUserReview(page, userId)` 를 호출해 입력받은 페이지에 대해 특정 사용자가 작성한 리뷰를 보여준다.

```
/**      * Retrieves a page of reviews for a specific user from the database.
 * @param scanner a Scanner object to read user input      */    public
static void getUserReview(Scanner scanner) {      System.out.print("Enter
user ID: ");      String userId = scanner.nextLine();      System.out.pr
int("Enter page number: ");      int page = scanner.nextInt();      scan
ner.nextLine();
ArrayList<DB2024TEAM07_ResReviewVO> userReviews = reviewDAO.getUs
erReview(page, userId);      if (userReviews == null) {          System.o
ut.println("User ID not found.");      } else if (userReviews.isEmpty()) {
System.out.println("No reviews found for user " + userId);      } else {
System.out.printf("%-10s%-15s%-20s%-15s%-100s%n", "Review ID", "User ID",
"R
estaurant", "Rating", "Review Comment");      System.out.println("-----
-----");
for (DB2024TEA
M07_ResReviewVO review : userReviews) {          System.out.printf("%-
10d%-15s%-20s%-10d%-60s%n",
review.getReview_id(),
review.getUser_id(),
review.getRes_name(),
review.getReview_content());
}      } }
```

- `getResCount(Scanner scanner)`

- 입력받은 식당에 대한 총 리뷰 수를 검색하는 함수이다.

- 작동 방식:

1. 식당 ID를 입력받는다.

2. `reviewDAO.getResCount(userId)` 를 호출해 특정 식당에 대해 작성된 총 리뷰 수를 보여준다.

```
/**      * Retrieves the total number of reviews for a specific restaurant.
 * @param scanner a Scanner object to read user input      */    public
static void getResCount(Scanner scanner) {      System.out.print("Enter re
staurant ID: ");      int resId = scanner.nextInt();
int count = reviewDAO.getResCount(resId);      if (count == -1) {
System.out.println("Restaurant ID not found.");      } else if (count == -
2) {      System.out.println("An error occurred while retrieving dat
```

```
a."); } else { System.out.println("Total number of reviews
for restaurant " + resId + ": " + count); }
}
```

- `getResReview(Scanner scanner)`
  - 입력받은 식당에 대한 리뷰 페이지를 검색해 리뷰를 보여주는 함수이다.
  - 작동 방식:
    1. 식당 ID를 입력받는다.
    2. `reviewDAO.getResReview(page, userId)` 를 호출해 입력받은 페이지에 대해 특정 식당에 대해 작성된 리뷰를 보여준다.

```
/**      * Retrieves a page of reviews for a specific restaurant from the dat
abase.      * @param scanner a Scanner object to read user input      */
public static void getResReview(Scanner scanner) { System.out.print
("Enter restaurant ID: "); int resId = scanner.nextInt();
ArrayList<DB2024TEAM07_UserReview> restaurantReviews = reviewDAO.get
ResReview(page, resId); if (restaurantReviews == null) { S
ystem.out.println("Restaurant ID not found."); } else if (restaurantR
eviews.isEmpty()) { System.out.println("No reviews found for rest
aurant " + resId); } else { System.out.printf("%-10s%-15s%
-15s%-10s%-100s%n", "Review ID", "User ID", "User Name", "Rating", "Review C
ontent"); System.out.println("-----");
-----
-----");
for (DB2024TEAM07_UserReview review : restaur
antReviews) {
System.out.printf("%-10d%-15s%-13s%-10d%-60s%
n",
review.getReview_id(),
review.getName(),
review.getReview_content());
}
}
}
```

- `deleteReview(Scanner scanner)`
  - 리뷰와 평점을 삭제하는 함수이다.
  - 작동 방식:
    1. 삭제할 리뷰의 ID를 입력받는다.
    2. 트랜잭션을 시작한다.
    3. `reviewDAO.delete(reviewId)` 를 호출해 리뷰를 삭제한다. `DELETE FROM DB2024_Rating WHERE review_id = ?` 쿼리를 사용해 평점을 삭제한다.
    4. `getResIdForReview(conn, reviewId)` 로 식당 ID를 검색해 `ratingDAO.getAvg(resId)` 로 새로운 평균 평점을 계산하고 `UPDATE DB2024_Restaurant SET rating = ? WHERE res_id = ?` 쿼리를 사용해 식당의 평점으로 업데이트한다.
    5. 모든 작업이 성공하면 트랜잭션을 커밋하고, 그렇지 않으면 롤백한다.

```
/**      * Deletes a review from the database.      * @param scanner a Scan
ner object to read user input      */
public static void deleteReview(Scanner scanner) { System.out.print("Enter review ID to delete: ");
int r
eviewId = scanner.nextInt();
Connection conn = null; try { conn = DB2024TEAM07_Dat
abase.getInstance().getConnection(); conn.setAutoCommit(false);
int result = reviewDAO.delete(reviewId); if (result > 0)
{
System.out.println("Review deleted successfully.");
String ratingDeleteQuery = "DELETE FROM DB2024_Rating WHERE rev
iew_id = ?"; try { PreparedStatement pStmt =
conn.prepareStatement(ratingDeleteQuery); pStmt.setInt(1, re
viewId); pStmt.executeUpdate(); } catch (SQLException e)
{ e.printStackTrace(); }
}
}
}
```

```

        e.printStackTrace();
    }

    int resId = getResIdForReview(conn, reviewId);
    double newAvgRating = ratingDAO.getAvg(resId);

    String updateRatingQuery = "UPDATE DB2024_Restaurant SET rating = ? WHERE res_id = ?";
    try {
        PreparedStatement pStmt = conn.prepareStatement(updateRatingQuery);
        pStmt.setInt(1, resId);
        pStmt.setDouble(2, newAvgRating);
        pStmt.executeUpdate();
    } catch (SQLException e) {
        e.printStackTrace();
    }
    conn.commit();
} else {
    System.out.println("Failed to delete review.");
} catch (SQLException e) {
    if (conn != null) {
        conn.rollback();
    }
    ex.printStackTrace();
}
e.printStackTrace();
} finally {
    if (conn != null) {
        try {
            conn.setAutoCommit(true);
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}

```

- `getResIdForReview(Connection conn, int reviewId)`
  - 리뷰와 연결된 식당의 ID를 검색하는 helper method이다.
  - 작동 방식:
    1. `SELECT res_id FROM DB2024_Rating WHERE review_id = ?` 쿼리를 사용 식당 ID를 검한다.
    2. `pStmt` 객체를 사용하여 쿼리의 파라미터를 설정한다.
    3. `pStmt.executeQuery()` 메소드를 호출하여 쿼리를 실행하고, 결과를 반환한다.

```

/**
 * Helper method to retrieve the restaurant ID associated with a review
 * from the database.      * @param conn a Connection object representing the
 * database connection      * @param reviewId the ID of the review      * @return th
 * e restaurant ID or -1 if not found      */
private static int getResIdForReview(Connection conn, int reviewId) {
    String resIdQuery = "SELECT res_id F
ROM DB2024_Rating WHERE review_id = ?";
    try {
        PreparedStatement pStmt = conn.prepareStatement(resIdQuery);
        pStmt.setInt(1, reviewId);
        ResultSet rs = pStmt.executeQuery();
        if (rs.next()) {
            return rs.getInt("res_id");
        } else {
            return -1;
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

```

## ▼ UserManager

: 로그인/로그아웃, 회원가입, 유저 정보 검색, 업데이트 기능을 관리하는 클래스이다.

### 필드 설명

`userDAO` : `DB2024TEAM07_UserDAO` 객체로, 사용자 데이터 접근 계층과 상호작용한다.

`loggedInUser` : 현재 로그인된 사용자 정보를 담고 있는 `DB2024TEAM07_User` 객체이다.

- `login(String username, String password)`
  - 사용자가 입력한 아이디와 비밀번호로 로그인을 시도하는 메서드이다.
  - 작동 방식:
    1. `userDAO.signIn()` 메서드를 호출하여 로그인 시도 결과를 확인한다.
    2. 로그인 성공 시, `loggedInUser` 필드를 해당 사용자 정보로 설정하고 `true`를 반환한다.
    3. 로그인 실패 시, `false`를 반환한다.

```

public boolean login(String username, String password) {
    int loginResult = userDAO.signIn(username, password);
    if (loginResult == 1) {
        loggedInUser = userDAO.getUser(username);
        return true;
    }
    return false;
}

```

- `logout()`

- 현재 로그인된 사용자를 로그아웃하는 메서드이다.
- 작동 방식: `loggedInUser` 필드를 `null`로 설정한다.

```

public void logout() {
    loggedInUser = null;
}

```

- `addUser(DB2024TEAM07_User user)`

- 새로운 사용자를 데이터베이스에 추가하는 메서드이다.
- 작동 방식: `userDAO.add()` 메서드를 호출하여 사용자를 추가하고, 성공 여부를 반환한다.

```

public boolean addUser(DB2024TEAM07_User user) {
    return userDAO.add(user) > 0;
}

```

- `showMyInfo()`

- 현재 로그인된 사용자의 정보를 출력하는 메서드이다.
- 작동 방식: `userDAO.getUser()` 메서드를 호출하여 사용자 정보를 가져오고 출력한다.

```

public static void showMyInfo() {
    DB2024TEAM07_User user = userDAO.getUser(loggedInUser.getUser_id());
    System.out.println("\n==== My Information ====\n");
    System.out.println("ID: " + user.getUser_id());
    System.out.println("Name: " + user.getName());
    System.out.println("Student ID: " + user.getStudent_id());
    System.out.println("Email: " + user.getEmail());
    System.out.println("Location: " + user.getLocation());
    System.out.println("=====\n");
}

```

- `searchOtherUser(Scanner scanner)`

- 다른 사용자를 ID로 검색하는 메서드이다.
- 작동 방식:
  1. 사용자 ID를 입력받는다.
  2. `userDAO.getOtherUser()` 메서드를 호출하여 해당 사용자 정보를 가져온다.
  3. 사용자 정보가 존재하면 출력하고, 존재하지 않으면 사용자 정보를 찾을 수 없다고 출력한다.

```

public static void searchOtherUser(Scanner scanner) {
    System.out.print("Enter the user ID: ");
    String userId = scanner.nextLine();
    DB2024TEAM07_UserVO otherUser = userDao.getOtherUser(userId);

    if (otherUser != null) {
        System.out.println("\n== User Information ==\n");
        System.out.println("ID: " + otherUser.getUser_id());
        System.out.println("Email: " + otherUser.getEmail());
        System.out.println("\n=====");
    } else {
        System.out.println("The user could not be found..");
    }
}

```

- `deleteAccount(Scanner scanner)`

- 사용자가 자신의 계정을 삭제할 수 있도록 하는 메서드이다.
- 작동 방식:
  1. 계정 삭제 여부를 확인한다.
  2. 사용자가 'y'를 입력하면 비밀번호를 입력받아 계정을 삭제하고, 'n'을 입력하면 삭제를 취소한다.
  3. 비밀번호가 일치하지 않으면 삭제가 실패했음을 알린다.

```

public static void deleteAccount(Scanner scanner) {
    System.out.print("\nAre you sure you want to delete your account?\n");
    System.out.print("All data related to you will be deleted.\n\n");
    System.out.print("Enter 'y' if you want to delete, or 'n': ");
    String confirm = scanner.nextLine().toLowerCase();

    if (confirm.equals("y")) {
        System.out.print("Please enter your PASSWORD: ");
        String password = scanner.nextLine();
        int result = userDao.delete(loggedInUser.getUser_id(), password);

        if (result > 0) {
            System.out.println("Membership withdrawal completed.");
            DB2024TEAM07_Main.main(null);
            System.exit(0);
        } else {
            System.out.println("The password does not match.");
        }
    } else if (confirm.equals("n")) {
        System.out.println("Membership withdrawal canceled.");
    } else {
        System.out.println("Invalid input. Membership withdrawal canceled.");
    }
}

```

- `update(Scanner scanner)`

- 사용자가 자신의 정보를 업데이트할 수 있도록 하는 메서드이다.
- 작동 방식:

1. 새 비밀번호, 이름, 학번, 이메일, 위치를 입력받는다.
2. 입력값이 비어 있으면 기존 값을 유지한다.
3. 업데이트된 사용자 정보를 `userDAO.update()` 메서드를 통해 데이터베이스에 저장하고, 성공 여부를 출력한다.

```

public static void update(Scanner scanner) {
    System.out.println("\n===== Update My information =====\n");

    // 유저 아이디는 변경 불가, 현재 로그인된 유저의 아이디 사용
    String userId = loggedInUser.getUser_id();

    System.out.print("New Password (Press enter to skip): ");
    String newUserPw = scanner.nextLine();
    String userPw = newUserPw.isEmpty() ? loggedInUser.getUser_pw() : newUserPw;

    System.out.print("New Name (Press enter to skip): ");
    String newName = scanner.nextLine();
    String name = newName.isEmpty() ? loggedInUser.getName() : newName;

    System.out.print("New Student ID (Press enter to skip): ");
    String newStudentIdStr = scanner.nextLine();
    int studentId = newStudentIdStr.isEmpty() ? loggedInUser.getStudent_id() : Integer.parseInt(newStudentIdStr);

    System.out.print("New Email (Press enter to skip): ");
    String newEmail = scanner.nextLine();
    String email = newEmail.isEmpty() ? loggedInUser.getEmail() : newEmail;

    System.out.print("New Location (Press enter to skip): ");
    String newLocation = scanner.nextLine();
    String location = newLocation.isEmpty() ? loggedInUser.getLocation() : newLocation;

    DB2024TEAM07_User updatedUser = new DB2024TEAM07_User(userId, userPw, name, studentId, email, location);
    int result = userDAO.update(updatedUser, loggedInUser.getUser_id());

    if (result > 0) {
        System.out.println("Update successful!");
        loggedInUser = updatedUser; // 업데이트된 사용자 정보로 loggedInUser 업데이트
    } else {
        System.out.println("Failed to update information.");
    }
}

```

- `displayAllUsers()`
  - 데이터베이스에 있는 모든 사용자 계정을 표시하는 메서드이다.
  - 작동 방식:
    1. `userDAO` 인스턴스를 사용하여 모든 사용자 계정을 검색한다.
    2. 사용자 ID, 이름, 학번, 이메일, 위치를 출력한다.

```

/**
 * Displays a list of all users in the system.

```

```

/*
public static void displayAllUsers() {
    List<DB2024TEAM07_User> allUsers = userDAO.getAllUsers();

    System.out.println("\n==== All Users ====\n");
    System.out.printf("%-15s%-15s%-15s%-25s%-15s%n", "ID", "Name", "Student
ID", "Email", "Location");
    System.out.println("-----");
    for (DB2024TEAM07_User user : allUsers) {
        System.out.printf("%-15s%-15s%-15d%-25s%-15s%n",
                           user.getUser_id(), user.getName(), user.getStudent_id(),
                           user.getEmail(), user.getLocation());
    }
}

```

- addAccountByManager(Scanner scanner)
  - 관리자가 새 사용자 계정을 추가할 수 있도록 하는 메서드이다.
  - 작동 방식:
    1. 새 사용자 정보(ID, 비밀번호, 이름, 학번, 이메일, 위치)를 입력하도록 요청한다.
    2. 입력된 정보를 사용하여 `DB2024TEAM07_User` 객체를 생성한다.
    3. `userDAO` 인스턴스를 사용하여 새 사용자를 데이터베이스에 추가한다.
    4. 추가가 성공하면 "The new user has been successfully added." 메시지를 출력하고, 실패하면 "Failed to add the new user." 메시지를 출력한다.

```

/**
 * Allows managers to add new users.
 *
 * @param scanner a Scanner object to read user input
 */
public void addAccountByManager(Scanner scanner) {
    System.out.print("\nEnter the new user information:\n");
    System.out.print("ID: ");
    String userId = scanner.nextLine();
    System.out.print("Password: ");
    String password = scanner.nextLine();
    System.out.print("Name: ");
    String name = scanner.nextLine();
    System.out.print("Student ID: ");
    int studentId = Integer.parseInt(scanner.nextLine());
    System.out.print("Email: ");
    String email = scanner.nextLine();
    System.out.print("Location: ");
    String location = scanner.nextLine();

    // 새로운 사용자 객체 생성
    DB2024TEAM07_User newUser = new DB2024TEAM07_User(userId, password, nam
e, studentId, email, location);

    // DB2024TEAM07_UserDAO 객체 생성
    DB2024TEAM07_UserDAO userDAO = new DB2024TEAM07_UserDAO();

    // 사용자 추가 메서드 호출
    int success = userDAO.add(newUser);
}

```

```

    // 결과 출력
    if (success > 0) {
        System.out.println("The new user has been successfully added.");
    } else {
        System.out.println("Failed to add the new user.");
    }
}

```

- updateAccountByManager(Scanner scanner)
  - 관리자가 기존 사용자 계정 정보를 업데이트할 수 있도록 하는 메서드이다.
  - 작동 방식:
    1. 업데이트할 사용자의 ID를 입력하도록 요청한다.
    2. 새 비밀번호, 이름, 학번, 이메일, 위치를 입력하도록 요청한다 (값을 입력하지 않으면 기존 값으로 유지됨).
    3. 업데이트된 사용자 정보를 사용하여 DB2024TEAM07\_User 객체를 생성한다.
    4. userDAO 인스턴스를 사용하여 사용자 정보를 업데이트한다.
  - 5. 업데이트가 성공하면 "User information updated successfully!" 메시지를 출력하고, 실패하면 "Failed to update user information." 메시지를 출력한다.

```

/**
 * Allows managers to update existing user information.
 *
 * @param scanner a Scanner object to read user input
 */
public void updateAccountByManager(Scanner scanner) {
    System.out.print("\nEnter the ID of the user to update: ");
    String userId = scanner.nextLine();
    DB2024TEAM07_User userToUpdate = userDAO.getUser(userId);

    if (userToUpdate != null) {
        System.out.print("New Password (Press enter to skip): ");
        String newPassword = scanner.nextLine();
        String password = newPassword.isEmpty() ? userToUpdate.getUser_pw() : newPassword;

        System.out.print("New Name (Press enter to skip): ");
        String newName = scanner.nextLine();
        String name = newName.isEmpty() ? userToUpdate.getName() : newName;

        System.out.print("New Student ID (Press enter to skip): ");
        String newStudentIdStr = scanner.nextLine();
        int studentId = newStudentIdStr.isEmpty() ? userToUpdate.getStudent_id() : Integer.parseInt(newStudentIdStr);

        System.out.print("New Email (Press enter to skip): ");
        String newEmail = scanner.nextLine();
        String email = newEmail.isEmpty() ? userToUpdate.getEmail() : newEmail;

        System.out.print("New Location (Press enter to skip): ");
        String newLocation = scanner.nextLine();
        String location = newLocation.isEmpty() ? userToUpdate.getLocation() : newLocation;
    }

    // 업데이트된 사용자 객체 생성
    DB2024TEAM07_User updatedUser = new DB2024TEAM07_User(userId, passwor

```

```

d, name, studentId, email, location);

        // 사용자 정보 업데이트 메서드 호출
        int result = userDao.update(updatedUser, userId);

        if (result > 0) {
            System.out.println("User information updated successfully!");
        } else {
            System.out.println("Failed to update user information.");
        }
    } else {
        System.out.println("User not found.");
    }
}

```

- `searchAccountByManager(Scanner scanner)`

- 관리자가 사용자 ID로 특정 사용자를 검색할 수 있도록 하는 메서드이다.
- 작동 방식:
  1. 검색할 사용자의 ID를 입력하도록 요청한다.
  2. 입력된 ID에 해당하는 사용자 정보를 가져온다.
  3. 사용자 정보를 출력한다.

```

/**
 * Allows managers to search for users by ID.
 *
 * @param scanner a Scanner object to read user input
 */
public void searchAccountByManager(Scanner scanner) {
    System.out.print("\nEnter the ID of the user to search: ");
    String userId = scanner.nextLine();
    DB2024TEAM07_User user = userDao.getUser(userId);

    if (user != null) {
        System.out.println("\n==== User Information ====\n");
        System.out.printf("%-15s%-15s%-15s%-15s\n", "ID", "Name", "Student ID", "Email", "Location");
        System.out.println("-----");
        System.out.printf("%-15s%-15s%-15d%-25s%-15s\n",
                          user.getUser_id(), user.getName(), user.getStudent_id(),
                          user.getEmail(), user.getLocation());
        System.out.println("\n=====");
    } else {
        System.out.println("User not found.");
    }
}

```

- `deleteAccountByManager(Scanner scanner)`

- 관리자가 사용자 계정을 삭제할 수 있도록 하는 메서드이다.
- 작동 방식:
  1. 삭제할 사용자의 ID를 입력하도록 요청한다.
  2. 사용자 계정 삭제를 확인하는 메시지를 출력한다.

3. 사용자가 확인한 경우, 비밀번호를 입력하여 삭제를 확인한다.
4. `userDAO` 인스턴스를 사용하여 사용자 계정을 삭제한다.
5. 삭제가 성공하면 "The user account has been successfully deleted." 메시지를 출력하고, 비밀번호가 일치하지 않으면 "The password does not match. User account deletion canceled." 메시지를 출력한다. 그 외의 경우에는 "An error occurred while deleting the user account." 메시지를 출력한다.

```
/**
 * Allows managers to delete users.
 *
 * @param scanner a Scanner object to read user input
 */
public void deleteAccountByManager(Scanner scanner) {
    System.out.print("\nEnter the user ID you want to delete: ");
    String userId = scanner.nextLine();

    System.out.println("\nYou are about to delete a user account.");
    System.out.println("All data related to this user will be permanently deleted.");
    System.out.print("Are you sure you want to delete the account with ID '" + userId + "'? (y/n): ");
    String confirm = scanner.nextLine().toLowerCase();

    if (confirm.equals("y")) {
        System.out.print("Enter password to confirm: ");
        String password = scanner.nextLine();

        int result = userDAO.delete(userId, password);

        if (result > 0) {
            System.out.println("The user account has been successfully deleted.");
        } else if (result == 0) {
            System.out.println("The password does not match. User account deletion canceled.");
        } else {
            System.out.println("An error occurred while deleting the user account.");
        }
    } else if (confirm.equals("n")) {
        System.out.println("User account deletion canceled.");
    } else {
        System.out.println("Invalid input. User account deletion canceled.");
    }
}
```

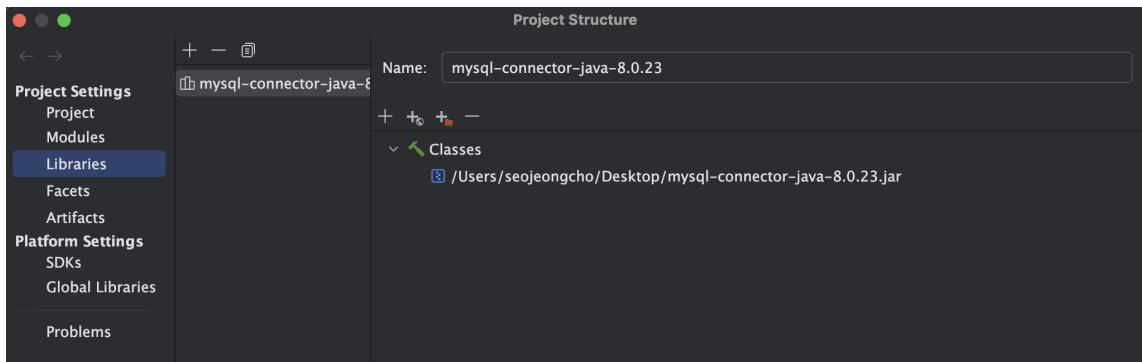
## ▼ 5. 응용프로그램 사용 방법

### MySQL에 사용자(DB2024TEAM07) 생성 쿼리 작성

```
CREATE USER DB2024TEAM07 IDENTIFIED BY 'DB2024TEAM07';
```

### JDBC와 MySQL 연결을 위한 환경 구성

- mysql-connector-java.jar 을 설치하고 File > Project Structure > Libraries 에서 해당 파일을 추가한다.



- 데이터베이스에 연결하기 위해 필요한 URL, 사용자 이름, 비밀번호를 설정한다.

```
static final String DB_URL = "jdbc:mysql://localhost:3306/DB2024TEAM07";
static final String USER = "DB2024TEAM07";
static final String PASS = "DB2024TEAM07";
```

- DriverManager.getConnection() 메서드를 사용하여 데이터베이스에 연결한다.

```
private DB2024TEAM07_Database() {
    try {
        conn = DriverManager.getConnection(DB_URL, USER, PASS);
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
```

## ▼ 6. 요구사항 분석

- (1) 5개 이상의 테이블을 가지고 있어야 하고 각 테이블들의 컬럼(Attribute)의 수를 합하면 20개 이상이어야 한다.

- 총 6개의 Table

```
mysql> SHOW TABLES;
+-----+
| Tables_in_db2024team07 |
+-----+
| DB2024_Menu
| DB2024_Rating
| DB2024_Restaurant
| DB2024_Review
| DB2024_Review_Menu_Res_Mapping
| DB2024_User
+-----+
6 rows in set (0.01 sec)
```

- 총 25개의 Attribute
  - DB2024\_Menu: 4개

```
mysql> DESCRIBE DB2024_Menu;
+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+
menu_id	int	NO	PRI	NULL	
menu_name	varchar(50)	YES		NULL	
res_id	int	NO	PRI	NULL	
price	int	YES		NULL	
+-----+-----+-----+-----+-----+
4 rows in set (0.00 sec)
```

- DB2024\_Restaurant: 9개

```
mysql> DESCRIBE DB2024_Restaurant;
+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+
res_name	varchar(200)	NO		NULL	
res_id	int	NO	PRI	NULL	auto_increment
phone_num	varchar(20)	YES		NULL	
address	varchar(200)	YES		NULL	
operating_hours	varchar(100)	YES		NULL	
break_time	varchar(100)	YES		NULL	
rating	decimal(2,1)	YES		NULL	
cuisine_type	varchar(50)	YES		NULL	
location	varchar(50)	YES		NULL	
+-----+-----+-----+-----+-----+
9 rows in set (0.00 sec)
```

- DB2024\_User: 6개

```
mysql> DESCRIBE DB2024_User;
+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+
user_id	varchar(50)	NO	PRI	NULL	
user_pw	varchar(50)	NO		NULL	
name	varchar(50)	NO		NULL	
student_id	int	YES		NULL	
email	varchar(50)	YES		NULL	
location	varchar(100)	YES		NULL	
+-----+-----+-----+-----+-----+
6 rows in set (0.00 sec)
```

- DB2024\_Review: 4개

```
mysql> DESCRIBE DB2024_Review;
+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+
review_id	int	NO	PRI	NULL	auto_increment
user_id	varchar(50)	NO	MUL	NULL	
rating	int	NO		NULL	
review_content	varchar(500)	YES		NULL	
+-----+-----+-----+-----+-----+
4 rows in set (0.00 sec)
```

- DB2024\_Rating: 2개

```
mysql> DESCRIBE DB2024_Rating;
+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+
| review_id | int | NO | PRI | NULL | |
| res_id | int | NO | PRI | NULL | |
+-----+-----+-----+-----+-----+
```

- DB2024\_Review\_Menu\_Res\_Mapping: 4개

```
mysql> DESCRIBE DB2024_Review_Menu_Res_Mapping;
+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+
mapping_id	int	NO	PRI	NULL	auto_increment
review_id	int	YES	MUL	NULL	
menu_id	int	YES	MUL	NULL	
res_id	int	YES	MUL	NULL	
+-----+-----+-----+-----+-----+
4 rows in set (0.01 sec)
```

▼ (2) 초기화를 위해 적어도 30개의 레코드(튜플)를 가지고 있어야 한다. (모든 테이블들의 레코드 수의 총합이 30개 이상)

- 총 235개 레코드

- DB2024\_User: 5개

```
mysql> SELECT * FROM DB2024_User;
+-----+-----+-----+-----+-----+-----+
| user_id | user_pw | name | student_id | email | location |
+-----+-----+-----+-----+-----+-----+
astralfinance	astralfinance	한사랑	2271064	astralfinance@ewhain.net	후문
cannes7	cannes7	고은서	2122004	cannes7@ewhain.net	정문
chacha091	chacha091	차현주	2276321	chacha09@ewhain.net	정문
meanwest	meanwest	김민서	2276046	meanwestk@gmail.com	후문
s2eojeong	s2eojeong	조서정	2276305	s2eojeong@gmail.com	후문
+-----+-----+-----+-----+-----+
5 rows in set (0.00 sec)
```

- DB2024\_Menu: 75개

| mysql> SELECT * FROM DB2024_Menu; |                     |        |       |
|-----------------------------------|---------------------|--------|-------|
| menu_id                           | menu_name           | res_id | price |
| 1                                 | 육회덮밥                | 1      | 11000 |
| 1                                 | 사케동                 | 2      | 14000 |
| 1                                 | 감봉뵈르                | 3      | 11000 |
| 1                                 | 전복 관자 파스타           | 4      | 23000 |
| 1                                 | (스페인에서 온 올리브) 감바스새우 | 5      | 9400  |
| 1                                 | 등심돈까스               | 6      | 9500  |
| 1                                 | 붓가게                 | 7      | 7000  |
| 1                                 | 아코스톨김밥              | 8      | 3000  |
| 1                                 | 사케동                 | 9      | 15000 |
| 1                                 | 삼겹 샤브세트             | 10     | 12000 |
| 1                                 | 제육덮밥                | 11     | 8500  |
| 1                                 | 치킨라이스 보통            | 12     | 10000 |
| 1                                 | 모로코 칼류 플레이트         | 13     | 25000 |
| 1                                 | 우삼겹정식               | 14     | 16000 |
| 1                                 | 후띠우 쌀국수             | 15     | 9000  |
| 1                                 | 떡갈비 도시락             | 16     | 12000 |
| 1                                 | 매콤명란크림파스타           | 17     | 9500  |
| 1                                 | 베트남 치킨반미            | 18     | 7300  |
| 1                                 | 연잎밥 정식              | 19     | 18000 |
| 1                                 | 정년찌개                | 20     | 8000  |
| 1                                 | 파마산 치킨파스타           | 21     | 12000 |
| 1                                 | 베이징 가지덮밥            | 22     | 13500 |
| 1                                 | 불돼지                 | 23     | 10000 |
| 1                                 | 불고기정식               | 24     | 12000 |
| 1                                 | 러우지아모               | 25     | 4000  |
| 1                                 | 꿔바로우                | 26     | 12900 |
| 1                                 | 훈제연어 샌드위치           | 27     | 8900  |
| 1                                 | 짜장면                 | 28     | 5500  |
| 1                                 | 김치찌개                | 29     | 3000  |
| 1                                 | 얼큰 버섯 쌀국수           | 30     | 10000 |
| 1                                 | 스파이스 연어 포케          | 31     | 12500 |
| 2                                 | 소고기 가지덮밥            | 1      | 15000 |
| 2                                 | 간강새우계란밥             | 2      | 12000 |
| 2                                 | 허머스 라페 치아바타         | 3      | 9000  |
| 2                                 | 홍새우 파스타             | 4      | 23000 |
| 2                                 | (노르웨이 불곰도 반한) 생연어   | 5      | 9900  |
| 2                                 | 안심돈까스               | 6      | 11000 |
| 2                                 | 기초네                 | 7      | 7500  |
| 2                                 | 순대떡볶음               | 8      | 3900  |
| 2                                 | 아부리사 캐동             | 9      | 15000 |
| 2                                 | 버섯 샤브세트             | 10     | 12000 |
| 2                                 | 가라아게동               | 11     | 9000  |
| 2                                 | 치킨라이스 특             | 12     | 14000 |
| 2                                 | 터키 쿨브르 플레이트         | 13     | 23000 |
| 2                                 | 마구로정식               | 14     | 16000 |
| 2                                 | 분짜                  | 15     | 10000 |
| 2                                 | 소불고기 도시락            | 16     | 15000 |
| 2                                 | 알리오올리오              | 17     | 8500  |
| 2                                 | 베이컨 치즈 오믈렛          | 18     | 7800  |
| 2                                 | 곤드레밥 정식             | 19     | 16000 |
| 2                                 | 궁극의베이컨볶음밥           | 20     | 8500  |
| 2                                 | 토마토 해산물 파스타         | 21     | 12000 |
| 2                                 | 쉬림프 크림 파스타          | 22     | 16500 |
| 2                                 | 불오징어                | 23     | 12000 |
| 2                                 | 둘깨미역국               | 24     | 9500  |
| 2                                 | 비빔량피                | 25     | 6000  |
| 2                                 | 마라탕                 | 26     | 7000  |
| 2                                 | 낫소                  | 27     | 2800  |
| 2                                 | 짬뽕                  | 28     | 6500  |
| 2                                 | 라면사리                | 29     | 1000  |
| 2                                 | 맑은 버섯 쌀국수           | 30     | 10000 |
| 2                                 | 클래식 참치 포케           | 31     | 11500 |
| 3                                 | 스모크드 베이컨 할리피뇨 바게트   | 3      | 11000 |
| 3                                 | 흑돼지 핑크안심            | 4      | 17000 |
| 3                                 | 리코타치즈 샌드            | 5      | 7500  |
| 3                                 | 치즈돈까스               | 6      | 11000 |
| 3                                 | 가케                  | 7      | 6500  |
| 3                                 | 가츠동                 | 11     | 9000  |
| 3                                 | 쌈라펀                 | 25     | 7000  |
| 3                                 | 마라샹궈                | 26     | 15000 |
| 3                                 | 고기추가                | 29     | 2000  |
| 3                                 | 샤브소고기               | 30     | 10000 |
| 3                                 | 부채살 스테이크 보울         | 31     | 13500 |
| 4                                 | E.C.C 샐러드           | 4      | 16000 |
| 4                                 | 어묵사리                | 29     | 1000  |

- DB2024\_Restaurant: 31개

| res_name    | res_id | phone_num      | address                     | operating_hours                | break_time      | rating | cuisine_type | location |
|-------------|--------|----------------|-----------------------------|--------------------------------|-----------------|--------|--------------|----------|
| 오미자식당       | 1      | 070-4154-2890  | 서울 서대문구 이화여대길 24 2층         | 월~금 11:00~20:30, 토 11:00~20:30 | 월~토 15:00~17:00 | 4.0    | 일식           | 장문       |
| 남안식당        | 2      | 02-312-1238    | 서울 서대문구 이화여대길 6 1층          | 월~토 10:00~21:00                | 월~토 15:00~17:00 | 3.0    | 베이커리         | 장문       |
| 원조도시문       | 3      | 02-313-3198    | 서울 종로구 이화여대길 20 1층          | 월~금 11:00~20:30                | 월~금 15:00~17:00 | 3.0    | 한식           | 장문       |
| 조식카페트       | 4      | 02-363-0090    | 서울 서대문구 이화여대길 19 2층         | 월~금 11:00~20:30                | 월~금 15:00~17:00 | 4.0    | 조식카페         | 장문       |
| 조식카페        | 5      | 02-363-5679    | 서울 서대문구 이화여대길 19 1층         | 월~금 11:00~19:30                | 월~금 15:00~17:00 | 3.0    | 조식카페         | 장문       |
| 유아체육교       | 6      | 02-6441-1993   | 서울 서대문구 이화여대길 28 101호       | 월~토 11:00~21:00                | 월~토 15:00~17:00 | 3.0    | 유아체육교        | 장문       |
| 유아체육교       | 7      | 02-363-0090    | 서울 종로구 서대문구청 101호           | 월~금 11:00~20:30                | 월~금 15:00~17:00 | 5.0    | 유아체육교        | 장문       |
| 아몬스튜        | 8      | 02-364-1381    | 서울 종로구 신촌로 17 1층 110호       | 월~금 11:00~21:00                | 월~금 15:00~17:00 | 5.0    | 아몬스튜         | 장문       |
| 진동부리        | 9      | 010-4726-7684  | 서울 서대문구 신촌로 18 신촌 차이빌라 8184 | 월~토 17:00~20:30                | 월~토 15:00~17:00 | 4.0    | 일식           | 장문       |
| 어반소바사브      | 10     | 02-6442-4949   | 서울 서대문구 이화여대길 2 201호        | 월~금 11:00~20:30                | 월~금 15:00~17:00 | 4.0    | 소바사브         | 장문       |
| 어반소바사브      | 11     | 02-363-0090    | 서울 서대문구 이화여대길 19 1층         | 월~금 11:00~20:30                | 월~금 15:00~17:00 | 4.0    | 한식           | 장문       |
| 까이식당        | 12     | 070-7779-8899  | 서울 서대문구 이화여대길 24 1층         | 월~금 11:00~20:30                | 월~금 15:00~17:00 | 4.0    | 까이식당         | 장문       |
| 아민 이화       | 13     | 02-363-0113    | 서울 서대문구 이화여대길 52-1 1층       | 월~금 11:00~20:30                | 월~금 15:00~17:00 | 4.0    | 한식           | 장문       |
| 운영한정식당      | 14     | 02-363-0056    | 서울 서대문구 이화여대길 2 1층          | 월~금 11:00~20:30                | 월~금 15:00~17:00 | 4.0    | 한정식당         | 장문       |
| 운영한정식당      | 15     | 02-363-1985    | 서울 서대문구 이화여대길 2 1층          | 월~금 11:00~21:00                | 월~금 15:00~17:00 | 4.0    | 한정식당         | 장문       |
| 마더우 식신점     | 16     | 02-363-1157    | 서울 종로구 이화여대길 29 101호        | 월~금 9:00~18:00                 | 월~금 10:00~20:00 | 2.0    | 도시락          | 장문       |
| 마더우 식신점     | 17     | 0507-1413-0468 | 서울 서대문구 이화여대길 29 101호       | 월~금 10:00~20:00                | 월~금 11:00~20:00 | 4.0    | 한식           | 장문       |
| 터미널카페터      | 18     | 02-363-0053    | 서울 서대문구 이화여대길 29 101호       | 월~금 10:00~20:00                | 월~금 11:00~20:00 | 4.0    | 한식           | 장문       |
| 이로운 밥상      | 19     | 02-363-1245    | 서울 서대문구 연재동문길 27-6 2층       | 월~토 11:00~21:00                | 월~토 15:00~17:00 | 3.0    | 한식           | 후문       |
| 화가모아리서      | 20     | 02-364-1970    | 서울 서대문구 성산동 527 1층          | 월~금 11:00~20:00, 토 11:00~14:00 | 월~금 15:00~17:00 | 3.0    | 한식           | 후문       |
| 화가모아리서      | 21     | 02-363-0053    | 서울 서대문구 성산동 527 1층          | 월~금 11:00~20:00                | 월~금 15:00~17:00 | 3.0    | 한식           | 후문       |
| 식탁 하우슬점     | 22     | 02-363-0777    | 서울 서대문구 성산동 527 하우슬점        | 월~금 11:00~20:00                | 월~금 15:00~17:00 | 4.0    | 한식           | 후문       |
| 불법          | 23     | 02-362-9833    | 서울 서대문구 이화여대길 11 2층         | 월~토 11:00~22:00                | 월~토 15:00~16:30 | 3.0    | 한식           | 장문       |
| 불법          | 24     | 02-392-9333    | 서울 서대문구 이화여대길 11 2층         | 월~토 11:00~22:00                | 월~토 15:00~16:30 | 3.0    | 한식           | 장문       |
| 의왕점         | 25     | 02-363-0053    | 서울 서대문구 이화여대길 72-1 1층       | 월~금 10:30~21:30                | 월~금 11:00~21:30 | 4.0    | 한식           | 장문       |
| 화라운 마라탕     | 26     | 02-311-0158    | 서울 서대문구 이화여대길 72-1 1층       | 월~일 16:00~22:00                | 월~일 18:30~16:30 | 3.0    | 중식           | 장문       |
| 마더우 데리고글    | 27     | 070-7758-3870  | 서울 서대문구 이화여대길 72-1 1층       | 월~일 16:00~22:00                | 월~일 18:30~16:30 | 4.0    | 한식           | 장문       |
| 청년봉우리학교     | 28     | 0507-1344-6831 | 서울 서대문구 신촌로 43 1층           | 월~금 11:00~20:00                | 월~금 16:00~17:00 | 4.0    | 한식           | 장문       |
| 청년봉우리학교     | 29     | 0507-1398-7858 | 서울 서대문구 신촌로 43 1층           | 월~일 11:00~22:00                | 월~일 15:00~17:00 | 5.0    | 事业发展         | 장문       |
| 승촌소프트웨어 수신점 | 30     | 0507-1391-7188 | 서울 서대문구 이화여대길 78 가동 1층      | 월~일 11:00~21:00                | 월~일 15:00~17:00 | 4.0    | 샐러드          | 장문       |

31 rows in set (0.00 sec)

## o DB2024\_Review: 62개

| review_id | user_id       | rating | review_content                                                                  |
|-----------|---------------|--------|---------------------------------------------------------------------------------|
| 1         | s2eojeong     | 5      | 육회덮밥 너무 고소하고 맛있어요!!                                                             |
| 2         | cannes7       | 3      | 소고기 거지덮밥 맛있긴한데, 약간 비싸워ㅠㅠ                                                        |
| 3         | meanwest      | 5      | 여기 육회덮밥 진짜 너무 맛있어요.. 모모지 절대 지켜.                                                 |
| 4         | chacha091     | 5      | 사케동 정말 신선하고 맛있어요!                                                               |
| 5         | astralfinance | 5      | 간강새우계란밥이 진짜 최고예요... 눈을 훌리면서 먹었어요                                                |
| 6         | cannes7       | 4      | 가격대가 좀 높긴 해도 양이 많아서 먹고나면 배불리요ㅎㅎ                                                 |
| 7         | s2eojeong     | 0      | 스모크드 베이컨 할리피노 바게트 매콤하긴한데 맛있게 잘 먹었어요 추천!!                                        |
| 8         | meanwest      | 4      | 감본비로 둑특하고 맛있어요!                                                                 |
| 9         | astralfinance | 3      | 냥냥한 감본비로에요. 무난합니다.                                                              |
| 10        | meanwest      | 5      | 홍새우 파스타 바다 쟁 툴툴나고 맛있어요!                                                         |
| 11        | chacha091     | 4      | E.C.C 썰러드 원전 신선하고 무난하게 맛있어요!                                                    |
| 12        | s2eojeong     | 3      | 홍새우 저울을 먹어보는데 나쁘지 않아요.                                                          |
| 13        | meanwest      | 5      | 리코타치즈 들어온 샌드위치 꼭 드세요. 신세계입니다                                                    |
| 14        | cannes7       | 5      | 생연어 신선하고 촉촉하고 부드려워요ㅠㅠ 추천!!                                                      |
| 15        | chacha091     | 0      | 정말 불愠이 반할만하네요.                                                                  |
| 16        | astralfinance | 4      | 등심돈까스 무난하게 맛있어요~                                                                |
| 17        | cannes7       | 5      | 등심보다 부드러워서 먹기 좋아요.                                                              |
| 18        | meanwest      | 3      | 튀김 웃이 약간 높은데 내구성은 좋았어요.                                                         |
| 19        | astralfinance | 5      | 붓가게 진짜 맛있어요.. 꼭 드셔보세요 다들!                                                       |
| 20        | cannes7       | 4      | 기초네 치킨 들어보는데 역시 이 집은 다 맛있네요!                                                    |
| 21        | meanwest      | 5      | 가게를 시켜보는데 역시 이 집은 다 맛있네요!                                                       |
| 22        | chacha091     | 5      | 야쿠느트,김밥珉珉 가성비 좋고 맛있는 곳은 없어요.. 학부생들에게 한줄기의 빛.                                    |
| 23        | cannes7       | 5      | 가끔 문식 영길때 순대떡볶음 먹는데 여전히 맛있네요.                                                   |
| 24        | meanwest      | 4      | 바빠서 시간 없을때 간단하게 먹기 좋네요.                                                         |
| 25        | chacha091     | 5      | 사케동 위에 면이 너무 신선하고 부드러워요!                                                        |
| 26        | s2eojeong     | 4      | 학교 주변에 사케동 파는 곳 많았더니 여기도 나쁘지 않은 것 같아요.                                          |
| 27        | chacha091     | 5      | 아부리사케동도 맛있었네요. 불향이 베어져서 좋아요.                                                    |
| 28        | cannes7       | 3      | 언어가 너무 구워져서 안 촉촉했어요ㅠㅠ                                                           |
| 29        | chacha091     | 5      | 비오는 날 마다 삼겹살 먹으려고 가는데 참 맛있네요.                                                   |
| 30        | astralfinance | 5      | 버섯 샐러드 살겁사발만 맛있어요!                                                              |
| 31        | meanwest      | 4      | 제육덮밥.. 무난하게 맛있어요.                                                               |
| 32        | chacha091     | 5      | 등심보다 부드러워서 먹기 좋아요.                                                              |
| 33        | s2eojeong     | 4      | 튀김 웃이 약간 높은데 내구성이 좋네요.                                                          |
| 34        | cannes7       | 4      | 양이 약간 적긴해도 나쁘진 않네요.                                                             |
| 35        | chacha091     | 5      | 꽝소에도 자주 먹었는데 오늘은 특으로 시켰어요!                                                      |
| 36        | astralfinance | 4      | 생소한 이름인데 미국적인 맛이 있어요. 나쁘지 않아요!                                                  |
| 37        | chacha091     | 3      | 고기냄새가 약간 나는 것 같아요..                                                             |
| 38        | cannes7       | 5      | 위에 노른자 톡 터뜨려서 먹으면 맛있어요!                                                         |
| 39        | s2eojeong     | 5      | 돌이 먹다 하나 죽어도 모를 맛이에요.. 강추                                                       |
| 40        | cannes7       | 4      | 싱싱하니 맛있네요.                                                                      |
| 41        | s2eojeong     | 4      | 감시동안 브루노에 있다왔네요.                                                                |
| 42        | cannes7       | 4      | 국물이 깔끔하고 좋아요.                                                                   |
| 43        | meanwest      | 3      | 고기가 좀 펑펑해요.. 거기에 비해 고기양이 적은것도 같네요.                                              |
| 44        | chacha091     | 4      | 한식 맹꽁 때면 먹으면 좋아요.                                                               |
| 45        | astralfinance | 3      | 아무래도 소불고기가 가격대가 좀 있어요.. 맛있긴한데 아침ㅠㅠ                                              |
| 46        | meanwest      | 4      | 크림이면 브로꾸덕하기 맛인데 이건 매콤해서 느끼하지도 않고 좋아요. 강추!!                                      |
| 47        | cannes7       | 5      | 너무 맛있어요!!!! 단짠단짠의 대명사.                                                          |
| 48        | s2eojeong     | 5      | 파스타 중에서 오일 파스타를 제일 좋아하는데 맛있네요 증명                                                |
| 49        | cannes7       | 4      | 아침에 브루니치 먹으니까 맛있고 적당히 배부르네요.                                                    |
| 50        | meanwest      | 4      | 건강한 맛이라 같은 약하지만 그래도 먹을만해요.                                                      |
| 51        | cannes7       | 4      | msg 하나 없이 자연친화적인 맛이에요. 속세를 벗어나고 싶으신 분들께 추천드려요.                                  |
| 52        | s2eojeong     | 3      | 나물이 약간 질거서 씹는다 불편했어요ㅠㅠ                                                          |
| 53        | meanwest      | 4      | 엄마가 해주는 짜개 맛 나오요. 얼른 증강했으면                                                      |
| 54        | cannes7       | 4      | 집에서 해먹는 봄음밥 맛이에요. 나쁘지 않아요.                                                      |
| 55        | meanwest      | 5      | 나쁘진 않았는데 다음엔 안 시킬 것 같아요..                                                       |
| 56        | chacha091     | 5      | 여기 소스가 진짜 특이하고 맛있어요!!                                                           |
| 57        | astralfinance | 4      | 기억하니 맛있네요!!                                                                     |
| 58        | meanwest      | 4      | 가지 맹꽁이 약간 질기긴 했지만 소스가 맛있어서 괜찮았어요!!                                              |
| 59        | chacha091     | 4      | 소스가 풀이한데 묘하게 증강되었어요..                                                           |
| 60        | cannes7       | 4      | .HorizontalAlignment 되게 좋아하는는데 들깨 미역국이래서 시켜보았어요. 나쁘진 않지만 역시 그냥 미역국이 더 나은 것 같아요. |
| 61        | meanwest      | 5      | 국물이 꾸-득하고 고소해요. 간만에 맛있게 석사했어요.                                                  |
| 62        | astralfinance | 3      | 영념이 너무 달아요ㅠㅠ                                                                    |

62 rows in set (0.00 sec)

## o DB2024\_Rating: 62개

```
mysql> SELECT * FROM DB2024_Rating;
+-----+-----+
| review_id | res_id |
+-----+-----+
1	1
2	1
3	1
4	2
5	2
6	2
7	3
8	3
9	3
10	4
11	4
12	4
13	5
14	5
15	5
16	6
17	6
18	6
19	7
20	7
21	7
22	8
23	8
24	8
25	9
26	9
27	9
28	9
29	10
30	10
31	11
32	11
33	12
34	12
35	13
36	13
37	14
38	14
39	14
40	14
41	15
42	15
43	15
44	16
45	16
46	17
47	17
48	17
49	18
50	18
51	19
52	19
53	20
54	20
55	21
56	21
57	22
58	22
59	22
60	23
61	23
62	24
+-----+-----+
62 rows in set (0.00 sec)
```

▼ (3) 기본 키(primary key), 외래 키(foreign key), not null 제약조건(not null constraints)를 포함해야 한다

```
CREATE TABLE DB2024_User(
    user_id VARCHAR(50) NOT NULL,
    user_pw VARCHAR(50) NOT NULL,
    name VARCHAR(50) NOT NULL,
    student_id INT,
    email VARCHAR(50) CHECK (email LIKE '%@%'),
    location VARCHAR(100),

    PRIMARY KEY (user_id)
);

CREATE TABLE DB2024_Restaurant(
    res_name VARCHAR(200) NOT NULL,
    res_id INT NOT NULL AUTO_INCREMENT,
    phone_num VARCHAR(20) DEFAULT NULL,
    address VARCHAR(200) DEFAULT NULL,
    operating_hours VARCHAR(100) DEFAULT NULL,
    break_time VARCHAR(100) DEFAULT NULL,
    rating decimal(2,1) DEFAULT NULL,
    cuisine_type VARCHAR(50) DEFAULT NULL,
    location VARCHAR(50) DEFAULT NULL,

    PRIMARY KEY(res_id)
);
```

```

CREATE TABLE DB2024_Menu(
    menu_id INT NOT NULL,
    menu_name VARCHAR(50) NOT NULL,
    res_id INT NOT NULL,
    price INT NOT NULL,
    PRIMARY KEY(menu_id, res_id),
    FOREIGN KEY(res_id) REFERENCES DB2024_Restaurant(res_id) ON DELETE CASCADE
);

CREATE TABLE DB2024_Review (
    review_id INT AUTO_INCREMENT,
    user_id VARCHAR(50) NOT NULL,
    rating INT NOT NULL CHECK(rating>-1 AND rating<6),
    review_content VARCHAR(500),
    PRIMARY KEY (review_id),
    FOREIGN KEY (user_id) REFERENCES DB2024_User(user_id) ON DELETE CASCADE
);

CREATE TABLE DB2024_Rating (
    review_id INT,
    res_id INT NOT NULL,
    PRIMARY KEY (review_id, res_id),
    FOREIGN KEY(review_id) REFERENCES DB2024_Review(review_id) ON DELETE CASCADE,
    FOREIGN KEY(res_id) REFERENCES DB2024_Restaurant(res_id) ON DELETE CASCADE
);

```

```

CREATE TABLE DB2024_Review_Menu_Res_Mapping (
    mapping_id INT AUTO_INCREMENT PRIMARY KEY,
    review_id INT,
    menu_id INT,
    res_id INT,
    FOREIGN KEY (review_id) REFERENCES DB2024_Review(review_id) ON DELETE CASCADE,
    FOREIGN KEY (menu_id) REFERENCES DB2024_Menu(menu_id) ON DELETE CASCADE,
    FOREIGN KEY (res_id) REFERENCES DB2024_Restaurant(res_id) ON DELETE CASCADE
);

```

▼ (4) 적어도 2개의 뷰를 정의해야 한다. (레포트에 view를 사용한 이유를 설명)

```

/* 보안용 유저정보 확인 뷰(다른 유저의 이름과 이메일만 확인 가능) */
CREATE VIEW DB2024_OtherUser AS
(SELECT user_id, `name`, email
FROM DB2024_User);

/* 리뷰를 유저이름, 평점, 리뷰내용의 형태로 보기 위한 뷰 */
CREATE VIEW DB2024_UserReview AS
SELECT review_id, DB2024_User.name, rating, review_content
FROM DB2024_User, DB2024_Review
WHERE DB2024_User.user_id = DB2024_Review.user_id;

/* 리뷰를 식당이름, 평점, 리뷰내용의 형태로 보기 위한 뷰 */
CREATE VIEW DB2024_ResReview AS
SELECT r.review_id, user_id, DB2024_Restaurant.res_name, r.rating, review_content
FROM DB2024_Restaurant, DB2024_Review r, DB2024_Rating

```

```

WHERE DB2024_Restaurant.res_id = DB2024_Rating.res_id
AND r.review_id = DB2024_Rating.review_id;

```

1. **DB2024\_OtherUser**: 다른 유저들을 조회할 때 쓰이는 뷔

보안성: 사용자 별로 필요한 데이터만 확인해 볼 수 있게끔 유저 테이블의 개인정보(user\_pw, location)들은 볼 수 없게 처리했다.

2. **DB2024\_UserReview**: 특정 식당의 리뷰를 조회할 때 쓰이는 뷔

편리성: 자주 사용되는 복잡한 질의를 미리 정의해 사용했다.

사용자 편의성을 고려해 특정 식당의 리뷰 조회 시 정수값인 user\_id를 식당 이름인 name으로 변환해서 보여주는 뷔이다.

3. **DB2024\_ResReview**: 특정 유저의 리뷰를 조회할 때 쓰이는 뷔

편리성: 자주 사용되는 복잡한 질의를 미리 정의해 사용했다.

사용자 편의성을 고려해 특정 유저의 리뷰 조회 시 정수값인 res\_id를 식당 이름인 res\_name으로 변환해서 보여주는 뷔이다.

▼ (5) 모든 테이블과 뷔의 이름들은 “DB2024\_”라는 접두어를 가지고 있어야 한다.

- DB2024\_ : 테이블
- db2024\_ : 뷔

```

mysql> SHOW TABLES;
+-----+
| Tables_in_db2024team07 |
+-----+
| DB2024_Menu
| db2024_menuview
| db2024_otheruser
| DB2024_Rating
| db2024_resreview
| DB2024_Restaurant
| DB2024_Review
| DB2024_Review_Menu_Res_Mapping
| DB2024_User
| db2024_userreview
+-----+
10 rows in set (0.00 sec)

```

▼ (6) 적어도 4개의 인덱스를 정의해야 한다.

```

/* DB2024_Rating.res_id: 특정 가게의 평점 평균을 구할 때 DB2024_Rating 테이블의 res_id가 자주 사용됨 */
CREATE INDEX DB2024_idx_AvgRating
    ON DB2024_Rating (res_id);

/* DB2024_Review.user_id: 특정 유저의 리뷰를 몰아볼 때 DB2024_Review 테이블의 user_id가 자주 사용됨 */
CREATE INDEX DB2024_idx_Review
    ON DB2024_Review (user_id);

/* DB2024_Menu.res_id: Restaurant별로 menu 검색 */
CREATE INDEX DB2024_idx_Menu
    ON DB2024_Menu(res_id);

/* DB2024_Restaurant.cuisine_type: cuisine_type별로 Restaurant를 검색 */
CREATE INDEX DB2024_idx_Restaurant
    ON DB2024_Restaurant(cuisine_type);

```

### 1. DB2024\_idx\_AvgRating

- : DB2024\_Rating 테이블의 res\_id 속성에 대한 인덱스다.
- 특정 가게의 평점 평균을 구할 때 res\_id 속성이 WHERE 절에서 자주 참조된다.
  - DB2024\_ResReview 생성 시에도 res\_id가 사용된다.
  - res\_id의 선택도는 상대적으로 낫다.

### 2. DB2024\_idx\_Review

- : DB2024\_Review 테이블의 user\_id 속성에 대한 인덱스다.
- 특정 유저의 리뷰를 찾아볼 때 user\_id가 WHERE 절에서 자주 참조된다.
  - DB2024\_UserReview 생성 시에도 user\_id가 사용된다.
  - user\_id의 선택도는 상대적으로 낫다.

### 3. DB2024\_idx\_Menu

- : DB2024\_Menu 테이블의 res\_id 속성에 대한 인덱스다.
- 특정 가게의 메뉴들만 찾아볼 때 res\_id 속성이 WHERE 절에서 자주 참조된다.

### 4. DB2024\_idx\_Restaurant

- : DB2024\_Restaurant 테이블의 cuisine\_type 속성에 대한 인덱스다.
- 특정 cuisine\_type별로 식당을 조회하고 싶을 때 cuisine\_type이 WHERE 절에서 자주 사용된다.

## ▼ (7) 인덱스를 사용하는 쿼리들을 포함해야 한다.

### 1. DB2024\_idx\_AvgRating

- 아래 함수에서 DB2024\_Rating 테이블 내 특정 res\_id 값을 가진 투플 총 개수를 조회할 때 사용된다.
- 쿼리문 내에서 명시적으로 DB2024\_idx\_AvgRating 인덱스 힌트를 주어서 res\_id 를 기준으로 테이블을 탐색하게 했다.

```
// src.com.database.DB2024TEAM07_RatingDAO.getAvg()

public float getAvg(int res_id){
    /*
        SELECT AVG(r.rating)
        FROM DB2024_Review r
        INNER JOIN DB2024_Rating ra
        USE INDEX (DB2024_idx_AvgRating) ON r.review_id = ra.review_id
        WHERE ra.res_id = ?
    */

    String Q = "SELECT AVG(r.rating) FROM DB2024_Review r INNER JOIN DB2024_Rating r
a USE INDEX (DB2024_idx_AvgRating) ON r.review_id = ra.review_id WHERE ra.res_id = ?";
    try{
        pStmt = conn.prepareStatement(Q);
        pStmt.setInt(1, res_id);
        rs = pStmt.executeQuery();
        if(rs.next()){
            return rs.getFloat(1);
        }
        return -1; //res_id is wrong
    }catch(SQLException se){
        se.printStackTrace();
    }
    return -2; //error
}
```

## 2. DB2024\_idx\_Review

- 아래 함수에서 `DB2024_Review` 테이블 내 특정 `user_id` 값을 가진 투플 총 개수를 조회할 때 사용된다.
- 쿼리문 내에서 명시적으로 `DB2024_idx_Review` 인덱스 힌트를 주어서 `user_id`를 기준으로 테이블을 탐색하게 했다.

```
// src.com.database.DB2024TEAM07_ReviewDAO.getUserCount()

public int getUserCount(String user_id) {
    /*
        SELECT COUNT(*)
        FROM DB2024_Review
        USE INDEX (DB2024_idx_Review)
        WHERE user_id = ?;
    */

    String Q = "SELECT COUNT(*) FROM DB2024_Review USE INDEX (DB2024_idx_Review) WHERE user_id = ?;";
    try{
        pStmt = conn.prepareStatement(Q);
        pStmt.setString(1, user_id)
        rs = pStmt.executeQuery();
        if(rs.next()) {
            return rs.getInt(1);
        }
        return 0; //아무 리뷰도 없는 상태
    }
    catch(SQLException se) {
        se.printStackTrace();
    }
    return -2; //error
}
```

## 3. DB2024\_idx\_Menu

- 한 식당 내의 존재하는 메뉴들을 조회할 때 사용된다.
- `DB2024_idx_Menu` 인덱스를 사용하여 `res_id`를 기준으로 메뉴를 검색한다.
- 인덱스는 `DB2024_Menu` 테이블에서 `res_id` 컬럼에 대해 생성되었기 때문에, 해당 컬럼을 조건으로 하는 쿼리에서 인덱스를 사용하여 검색 속도를 향상시킨다.

```
// src.com.database.DB2024TEAM07_MenuDAO.searchMenuByRestaurant()

public ResultSet searchMenuByRestaurant(int res_id) { // 메뉴 조회 - 식당별로 메뉴 검색
    String Q = "SELECT menu_id, menu_name, price FROM DB2024_Menu use index(DB2024_idx_Menu) WHERE res_id = ?;";
    try {
        pStmt = conn.prepareStatement(Q);
        pStmt.setInt(1, res_id);
        return pStmt.executeQuery();
    } catch (SQLException se) {
        se.printStackTrace();
    }
    return null;
}
```

## 4. DB2024\_idx\_Restaurant

- `cuisine_type` 별로 식당을 조회할 때 사용된다.
- `DB2024_idx_Restaurant` 인덱스를 사용하여 `cuisine_type`을 기준으로 식당을 검색한다.

- `DB2024_Restaurant` 테이블에서 `cuisine_type` 컬럼에 대해 생성되었기 때문에, `cuisine_type`을 조건으로 하는 쿼리에서 전체 테이블을 스캔하지 않고도 인덱스를 통해 해당 컬럼을 조건으로 하는 쿼리에서 인덱스를 사용하여 검색 속도를 향상시킨다.

```
// src.com.database.DB2024TEAM07_RestaurantDAO.searchRestaurantByCategory

public ResultSet searchRestaurantByCategory(String cuisine_type) {
    String Q = "SELECT res_name, phone_num, address, operating_hours, break_time, rating, location " +
               "FROM DB2024_Restaurant USE INDEX(DB2024_idx_Restaurant) WHERE cuisine_type = ?";
    try {
        pStmt = conn.prepareStatement(Q);
        pStmt.setString(1, cuisine_type);
        return pStmt.executeQuery();
    } catch (SQLException se) {
        se.printStackTrace();
    }
    return null;
}
```

#### ▼ (8) 뷰를 사용하는 쿼리를 포함해야 한다.

##### 1. DB2024\_OtherUser

- 아래 코드는 특정 레스토랑에 대한 유저들의 리뷰 정보를 반환하는 함수의 일부이다.
- 쿼리문에 의해 조회되는 테이블에 다른 유저의 `user_pw` 등의 정보까지 포함되는 것을 막기 위하여, `DB2024_OtherUser` 뷰를 조인하여 사용했다.

2. DB2024\_UserReview: 위 쿼리문에는 사용되지 않았지만, 해당 뷰를 사용하는 방식으로도 보안성을 충족할 수 있다.

```
//src.com.jdbc.database.DB2024TEAM07_ReviewDAO.getResReview()
public ArrayList<DB2024TEAM07_UserReview> getResReview(int page, int res_id){
/*
    SELECT *
    FROM DB2024_Review r NATURAL JOIN DB2024_OtherUser u
    WHERE review_id IN
        (SELECT review_id
        FROM DB2024_Rating
        WHERE res_id = ?)
    ORDER BY review_id DESC;
*/
    String Q = "SELECT * FROM DB2024_Review r NATURAL JOIN DB2024_OtherUser u WHERE review_id IN (SELECT review_id FROM DB2024_Rating WHERE res_id = ?) ORDER BY review_id DESC";
    //...//
```

- 아래 코드는 다른 유저의 정보를 반환하는 함수의 일부이다.
- 쿼리문에 의해 조회되는 테이블에 다른 유저의 `user_pw` 등의 정보까지 포함되는 것을 막기 위하여, `DB2024_OtherUser` 뷰를 사용했다.

```
//src.com.jdbc.database.DB2024TEAM07_ReviewDAO.getResReview()
public DB2024TEAM07_UserVO getOtherUser(String user_id) {
/*
    SELECT *
    FROM DB2024_OtherUser
    WHERE user_id = ?;
*/
    String Q = "SELECT * FROM DB2024_OtherUser WHERE user_id = ?";
    //...//
```

## 2. DB2024\_ResReview

- 아래 코드는 특정 유저의 레스토랑들에 대한 리뷰 정보를 반환하는 함수의 일부이다.
- DB2024\_Review 테이블로 해당 정보를 조회할 경우 유저의 이름이 아니라, DB2024\_User 테이블의 PK이자 DB2024\_Review 테이블의 FK인 `user_id` 가 반환된다.
- 이를 방지하고, 유저의 이름값을 반환하기 위해 `DB2024_ResReview` 류를 사용했다.

```
//src.com.jdbc.database.DB2024TEAM07_ReviewDAO.getUserReview()
public ArrayList<DB2024TEAM07_ResReviewVO> getUserReview(int page, String user_id){
/*
    SELECT *
    FROM DB2024_ResReview
    WHERE user_id= ?
    ORDER BY review_id DESC
*/
    String Q = "SELECT * FROM DB2024_ResReview WHERE user_id= ? ORDER BY review_id DESC";
    //...//
```

### ▼ (9) 트랜잭션(transaction)을 포함해야 한다.

#### 1. DB2024TEAM07\_ReviewManager

- 아래 코드는 사용자가 입력한 리뷰와 평점을 DB에 추가하고, 평균 평점을 업데이트하는 `addReview` 함수이다.
- 트랜잭션을 사용하여 리뷰, 평점 추가, 업데이트 작업을 하나의 논리적인 단위로 처리한다. 변경 사항을 커밋해 DB에 반영하고, 오류 발생 시 롤백하여 모든 변경 사항을 취소한다.
- 트랜잭션을 통해 DB의 일관성과 무결성을 유지한다.

```
// src.com.database.DB2024TEAM07_ReviewManager.addReview

public static void addReview(Scanner scanner) {
    System.out.print("Enter User ID: ");
    String userId = scanner.nextLine();

    System.out.print("Enter Restaurant ID: ");
    int resId = scanner.nextInt();

    System.out.print("Enter Menu ID: ");
    int menuId = scanner.nextInt();

    System.out.print("Enter Rating (1-5): ");
    int rating = scanner.nextInt();

    System.out.print("Enter review comment: ");
    scanner.nextLine();
    String reviewContent = scanner.nextLine();

    Connection conn = null;
    try {
        conn = DB2024TEAM07_Database.getInstance().getConnection();
        conn.setAutoCommit(false);

        DB2024TEAM07_Review review = new DB2024TEAM07_Review(0, userId, rating, reviewContent);
        int result = reviewDAO.add(review, menuId, resId);

        if (result > 0) {
            System.out.println("Review added successfully.");
            int reviewId = review.getReview_id();
        }
    } catch (Exception e) {
        e.printStackTrace();
        System.out.println("Error occurred while adding review.");
    } finally {
        if (conn != null) {
            try {
                conn.close();
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }
    }
}
```

```

        int ratingResult = ratingDAO.add(reviewId, resId);

        if (ratingResult > 0) {
            double newAvgRating = ratingDAO.getAvg(resId);
            if (newAvgRating >= 0) { // Ensure newAvgRating is valid
                String updateRatingQuery = "UPDATE DB2024_Restaurant SET rating
= ? WHERE res_id = ?";

                try (PreparedStatement pStmt = conn.prepareStatement(updateRating
Query)) {
                    pStmt.setDouble(1, newAvgRating);
                    pStmt.setInt(2, resId);
                    pStmt.executeUpdate();
                    conn.commit();
                    System.out.println("Restaurant rating updated successfull
y.");
                }
            } else {
                System.out.println("Failed to retrieve new average rating.");
                conn.rollback();
            }
        } else {
            System.out.println("Failed to add rating.");
            conn.rollback();
        }
    } else {
        System.out.println("Failed to add review.");
        conn.rollback();
    }
} catch (SQLException e) {
    e.printStackTrace();
    try {
        if (conn != null) {
            conn.rollback();
        }
    } catch (SQLException ex) {
        ex.printStackTrace();
    }
}
} finally {
    if (conn != null) {
        try {
            conn.setAutoCommit(true);
            //conn.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
}
}

```

2. DB2024TEAM07\_ReviewManager

- 아래 코드는 리뷰 평점과 내용을 수정하고 변경된 평점을 평균 평점에 업데이트하는 `updateReview` 함수이다.
  - 트랜잭션을 사용하여 리뷰와 평점 업데이트를 하나의 논리적 단위로 처리한다. 두 작업 중 하나라도 실패할 경우 전체 작업을 롤백 한다.
  - 이를 통해 DB의 일관성과 무결성을 유지한다.

```
// src.com.database.DB2024TEAM07_ReviewManager.updateReview
```

```

public static void updateReview(Scanner scanner) {
    System.out.print("Enter review ID to update: ");
    int reviewId = scanner.nextInt();

    System.out.print("Enter rating (1-5) to update: ");
    int rating = scanner.nextInt();

    System.out.print("Enter review content to update: ");
    scanner.nextLine();
    String reviewContent = scanner.nextLine();

    Connection conn = null;
    try {
        conn = DB2024TEAM07_Database.getInstance().getConnection();
        conn.setAutoCommit(false);

        DB2024TEAM07_Review review = new DB2024TEAM07_Review(reviewId, null, rating, reviewContent);
        int result = reviewDAO.update(review);

        if (result > 0) {
            System.out.println("Review updated successfully.");
        }

        int resId = getResIdForReview(conn, reviewId);

        double newAvgRating = ratingDAO.getAvg(resId);
        String updateRatingQuery = "UPDATE DB2024_Restaurant SET rating = ? WHERE res_id = ?";
        try {
            PreparedStatement pStmt = conn.prepareStatement(updateRatingQuery);
            pStmt.setDouble(1, newAvgRating);
            pStmt.setInt(2, resId);
            pStmt.executeUpdate();
            System.out.println("Restaurant rating updated successfully.");
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    conn.commit();

} else {
    System.out.println("Failed to update review.");
}
} catch (SQLException e) {
    if (conn != null) {
        try {
            conn.rollback();
        } catch (SQLException ex) {
            ex.printStackTrace();
        }
    }
    e.printStackTrace();
} finally {
    if (conn != null) {
        try {
            conn.setAutoCommit(true);
            //conn.close();
        } catch (SQLException e) {

```

```
        e.printStackTrace();
    }
}
```

3. DB2024TEAM07\_ReviewManager

- 아래 코드는 입력받은 ID의 리뷰를 삭제하고 삭제한 리뷰에 해당하는 레스토랑의 새 평균 평점을 업데이트하는 `deleteReview` 함수이다.
  - 트랜잭션을 통해 리뷰 삭제와 평균 평점 업데이트를 하나의 논리적 단위로 처리한다. 두 작업 중 하나라도 실패할 경우 전체 작업을 롤백한다.
  - 이를 통해 DB의 일관성과 무결성을 유지한다. 즉, 모든 작업이 성공적으로 완료되었을 때만 변경 사항이 커밋되도록 보장하여 데이터베이스 상태를 안정적으로 유지한다.

```
// src.com.database.DB2024TEAM07_ReviewManager.deleteReview

public static void deleteReview(Scanner scanner) {
    System.out.print("Enter review ID to delete: ");
    int reviewId = scanner.nextInt();

    Connection conn = null;
    try {
        conn = DB2024TEAM07_Database.getInstance().getConnection();
        conn.setAutoCommit(false);

        int result = reviewDAO.delete(reviewId);
        if (result > 0) {
            System.out.println("Review deleted successfully.");
        }

        String ratingDeleteQuery = "DELETE FROM DB2024_Rating WHERE review_id = ?";
        try {
            PreparedStatement pStmt = conn.prepareStatement(ratingDeleteQuery);
            pStmt.setInt(1, reviewId);
            pStmt.executeUpdate();
        } catch (SQLException e) {
            e.printStackTrace();
        }

        int resId = getResIdForReview(conn, reviewId);

        double newAvgRating = ratingDAO.getAvg(resId);
        String updateRatingQuery = "UPDATE DB2024_Restaurant SET rating = ? WHERE res_id = ?";
        try {
            PreparedStatement pStmt = conn.prepareStatement(updateRatingQuery);
            pStmt.setDouble(1, newAvgRating);
            pStmt.setInt(2, resId);
            pStmt.executeUpdate();
        } catch (SQLException e) {
            e.printStackTrace();
        }

        conn.commit();
    } else {
        System.out.println("Failed to delete review.");
    }
}
```

```

        } catch (SQLException e) {
            if (conn != null) {
                try {
                    conn.rollback();
                } catch (SQLException ex) {
                    ex.printStackTrace();
                }
            }
            e.printStackTrace();
        } finally {
            if (conn != null) {
                try {
                    conn.setAutoCommit(true);
                } catch (SQLException e) {
                    e.printStackTrace();
                }
            }
        }
    }
}

```

▼ (10) 중첩된 쿼리(nested query)들을 가지는 쿼리들을 포함해야 한다.

1. DB2024TEAM07\_ReviewDAO

- 특정 레스토랑에 대한 리뷰를 반환하기 위해 중첩된 쿼리가 사용되었다.
- DB2024\_Rating(review\_id, res\_id)

```

//src.com.jdbc.database.DB2024TEAM07_ReviewDAO.getResReview()
public ArrayList<DB2024TEAM07_UserReview> getResReview(int page, int res_id){
/*
    SELECT *
    FROM DB2024_Review r NATURAL JOIN DB2024_OtherUser u
    WHERE review_id IN
        (SELECT review_id
        FROM DB2024_Rating
        WHERE res_id = ?)
    ORDER BY review_id DESC;
*/
    String Q = "SELECT * FROM DB2024_Review r NATURAL JOIN DB2024_OtherUser u WHERE review_id IN (SELECT review_id FROM DB2024_Rating WHERE res_id = ?) ORDER BY review_id DESC";
    //...
}

```

▼ (11) 조인 쿼리(join query)들을 가지는 쿼리들을 포함해야 한다.

1. DB2024TEAM07\_RatingDAO

- 아래 함수는 특정 레스토랑에 대한 rating 값의 평균을 DB2024\_Review 테이블에서 가져오는 함수의 일부이다.
- DB2024\_Review(review\_id, user\_id, rating, review\_content)와 DB2024\_Rating(review\_id, res\_id)을 review\_id에 대해 조인하여 식당의 총 평점을 얻어내는 쿼리문이 사용되었다.

```

//src.com.jdbc.database.DB2024TEAM07_RatingDAO.getAvg()
public float getAvg(int res_id){
/*
    SELECT AVG(r.rating)
    FROM DB2024_Review r
    INNER JOIN DB2024_Rating ra
    USE INDEX (DB2024_idx_AvgRating) ON r.review_id = ra.review_id
    WHERE ra.res_id = ?
*/
}

```

```
String Q = "SELECT AVG(r.rating) FROM DB2024_Review r INNER JOIN DB2024_Rating  
//...//
```

## 2. DB2024TEAM07\_ReviewDAO

- 아래 함수는 특정 레스토랑에 대한 유저들의 리뷰 정보를 반환하는 함수의 일부이다.
- `DB2024_Review` 와 `DB2024_OtherUser` 를 자연 조인하여 `DB2024_UserReview(user_id, review_id, rating, review_content, name, email)`의 형태로 반환하는 쿼리문이 사용되었다.

```
//src/com/jdbc/database/DB2024TEAM07_ReviewDAO.getResReview()  
public ArrayList<DB2024TEAM07_UserReview> getResReview(int page, int res_id){  
/*  
     SELECT *  
     FROM DB2024_Review r NATURAL JOIN DB2024_OtherUser u  
     WHERE review_id IN  
         (SELECT review_id  
         FROM DB2024_Rating  
         WHERE res_id = ?)  
     ORDER BY review_id DESC;  
*/  
    String Q = "SELECT * FROM DB2024_Review r NATURAL JOIN DB2024_OtherUser u WHERE  
//...//
```

▼ (12) 매개 변수를 가지면서 동적으로 만드는 쿼리를 포함해야 한다. 다시 말해, 사용자로부터 입력 값을 받고 사용자가 입력한 값으로 쿼리를 생성한다.

## 1. DB2024TEAM07\_MenuDAO

- a. `add(DB2024TEAM07_Menu menu)` : 관리자가 메뉴를 등록할 때 'menu\_id', 'menu\_name', 'res\_id', 'price'를 입력하면 해당 값으로 쿼리를 동적으로 생성한다.

```
//src/com/jdbc/database/DB2024TEAM07_MenuDAO.java  
public int add(DB2024TEAM07_Menu menu) {  
    String Q = "INSERT INTO DB2024_Menu (menu_id, menu_name, res_id, price) VALUES  
(?, ?, ?, ?);  
    try {  
        pStmt = conn.prepareStatement(Q);  
        pStmt.setInt(1, menu.getMenu_id());  
        pStmt.setString(2, menu.getMenu_name());  
        pStmt.setInt(3, menu.getRes_id());  
        pStmt.setInt(4, menu.getPrice());  
  
        return pStmt.executeUpdate();  
  
    } catch (SQLException se) {  
        se.printStackTrace();  
    }  
    return 0;  
}
```

## 2. DB2024TEAM07\_RatingDAO

- a. `add(int review_id, int res_id)` : 사용자가 리뷰를 등록하면, 해당 리뷰의 `review_id` 와 사용자가 입력한 `res_id` 를 받아 내부적으로 쿼리가 동작한다.
- b. `getAvg(int res_id)` : 사용자가 리뷰를 등록하면, 사용자가 입력한 `res_id` 를 받아 내부적으로 쿼리가 동작한다.

```
//src/com/jdbc/database/DB2024TEAM07_RatingDAO.java  
public class DB2024TEAM07_RatingDAO{  
//...//
```

```

private PreparedStatement pStmt;
    //...
public int add(int review_id, int res_id){
    String Q = "INSERT INTO DB2024_Rating VALUES (?, ?)";
    try{
        pStmt = conn.prepareStatement(Q);
        pStmt.setInt(1, review_id);
        pStmt.setInt(2, res_id);
        return pStmt.executeUpdate();
    }
    //...
public float getAvg(int res_id){
    String Q = "SELECT AVG(r.rating) FROM DB2024_Review r INNER JOIN DB2024_Rating
ra USE INDEX (DB2024_idx_AvgRating) ON r.review_id = ra.review_id WHERE ra.res_id =
?";
    try{
        pStmt = conn.prepareStatement(Q);
        pStmt.setInt(1, res_id);
        rs = pStmt.executeQuery();
        if(rs.next()){
            return rs.getFloat(1);
        }
    }
    //...

```

### 3. DB2024TEAM07\_RestaurantDAO

- a. `searchRestaurantByCategory(String cuisine_type)`: 사용자가 cuisine\_type 별로 식당 검색을 할 때 'cuisine\_type'를 입력하면 해당 값으로 쿼리를 동적으로 생성한다.

```

//src/com/jdbc/database/DB2024TEAM07_RestaurantDAO.java
public ResultSet searchRestaurantByCategory(String cuisine_type) {
    String Q = "SELECT res_name, phone_num, address, operating_hours, break_time,
rating, location " +
    "FROM DB2024_Restaurant USE INDEX(DB2024_idx_Restaurant) WHERE cuisin
e_type = ?";
    try {
        pStmt = conn.prepareStatement(Q);
        pStmt.setString(1, cuisine_type);
        return pStmt.executeQuery();
    } catch (SQLException se) {
        se.printStackTrace();
    }
    return null;
}

```

### 4. DB2024TEAM07\_ReviewDAO

- a. `update(DB2024TEAM07_Review review)`: 사용자가 리뷰를 수정할 때 입력하여 ReviewDTO 객체에 저장된 `review_id`, `review_content`, `rating` 값을 받아와 쿼리문을 생성한다.

```

//src/com/jdbc/database/DB2024TEAM07_ReviewDAO.java
public class DB2024TEAM07_ReviewDAO {
    //...
    private PreparedStatement pStmt;
    //...
    public int update(DB2024TEAM07_Review review){
        String Q = "UPDATE DB2024_Review SET rating=?, review_content=? WHERE revi
ew_id=?";
        try{

```

```

    pStmt = conn.prepareStatement(Q);
    pStmt.setInt(1, review.getRating());
    pStmt.setString(2, review.getReview_content());
    pStmt.setInt(3, review.getReview_id());
    return pStmt.executeUpdate();
}catch(SQLException se){
    se.printStackTrace();
}
return -2; //error
}
//...

```

## 5. DB2024TEAM07\_UserDAO

- a. `delete(String user_id, String user_pw)` : 사용자가 회원탈퇴 기능을 이용할 때 입력/전달된 `user_id` 값을 받아와 쿼리문을 생성한다.
- (  
`user_pw` 값은 회원정보 유무와 회원정보 일치 여부 확인을 위해 사용되는 값이다.)

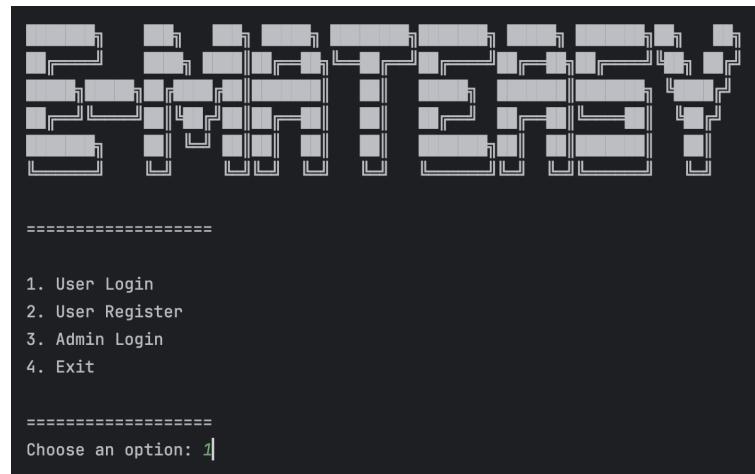
```

//src/com/jdbc/database/DB2024TEAM07_UserDAO.java
public class DB2024TEAM07_UserDAO {
    //...
    private PreparedStatement pStmt;
    //...
    public int delete(String user_id, String user_pw) {
        String Q = "DELETE FROM DB2024_User WHERE user_id = ?";
        try {
            int signInRes = signIn(user_id, user_pw);
            if (signInRes == 1) { //id ok pw ok
                pStmt = conn.prepareStatement(Q);
                pStmt.setString(1, user_id);
                return pStmt.executeUpdate(); // executeUpdate()로 변경
            } else { //0: id o pw x, -1: id x, -2: error
                return signInRes;
            }
        } catch (SQLException se) {
            se.printStackTrace();
        }
        return -2; //error
    }
    //...
}

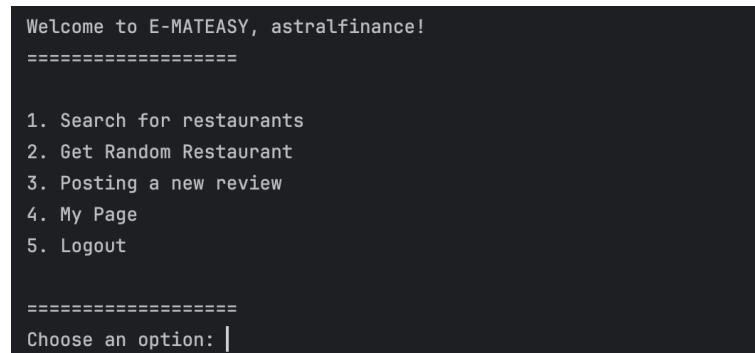
```

▼ (13) 그래픽 또는 문자 기반의 사용자 인터페이스를 사용해야 한다. 사용하기 쉽고 사용하기에 도움이 되는 정보를 가지고 있는 메뉴를 제공해야 한다.

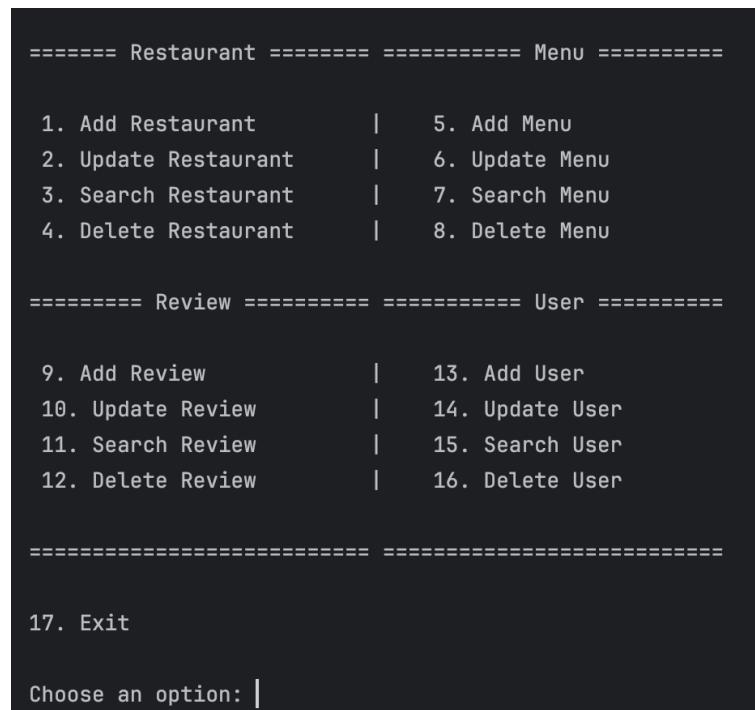
- 메인화면



- 유저 화면



- Admin 화면



- ▼ (14) 데이터베이스에 삽입(insert)을 하기 위한 인터페이스와 쿼리를 가지고 있어야 한다.

▼ 1. 회원가입

```
=====  
1. User Login  
2. User Register  
3. Admin Login  
4. Exit  
  
=====  
Choose an option: 2  
New username: ewha123  
New password: 1886  
Name: 김이화  
Student ID: 1871086  
Email (format: example@domain.com): ewha123@ewhain.net  
1. 정문  
2. 후문  
Choose an Location (Enter 1 or 2): 1  
Registration successful!
```

- 사용자가 회원가입을 할 때에는 순서대로 userid, password, name, student ID, Email, Location을 입력해야한다.
- email의 경우 지정된 format대로 작성해야한다.

회원가입 전)

```
mysql> SELECT * FROM DB2024_User;  
+-----+-----+-----+-----+-----+-----+  
| user_id | user_pw | name | student_id | email | location |  
+-----+-----+-----+-----+-----+-----+  
astralfinance	astralfinance	한사랑	2271064	astralfinance@	후문
cannes7	cannes7	고은서	2122004	cannes7@ewhain.net	정문
chacha091	chacha091	차현주	2276321	chacha09@ewhain.net	정문
meanwest	meanwest	김민서	2276046	meanwestk@gmail.com	후문
s2eojeong	s2eojeong	조서정	2276305	s2eojeong@gmail.com	후문
+-----+-----+-----+-----+-----+-----+  
5 rows in set (0.00 sec)
```

회원가입 후)

```

Database changed
mysql> SELECT * FROM DB2024_User;
+-----+-----+-----+-----+-----+-----+
| user_id | user_pw | name | student_id | email | location |
+-----+-----+-----+-----+-----+-----+
astralfinance	astralfinance	한사랑	2271064	astralfinance@ewhain.net	후문
cannes7	cannes7	고은서	2122004	cannes7@ewhain.net	정문
chacha091	chacha091	차현주	2276321	chacha09@ewhain.net	정문
ewha123	1886	김이화	1871086	ewha123@ewhain.net	정문
meanwest	meanwest	김민서	2276046	meanwestk@gmail.com	후문
s2eojeong	s2eojeong	조서정	2276305	s2eojeong@gmail.com	후문
+-----+-----+-----+-----+-----+-----+
6 rows in set (0.01 sec)

```

## ▼ 2. 메뉴 추가

```

===== Restaurant ===== ====== Menu =====

1. Add Restaurant | 5. Add Menu
2. Update Restaurant | 6. Update Menu
3. Search Restaurant | 7. Search Menu
4. Delete Restaurant | 8. Delete Menu

===== Review ===== ====== User =====

9. Add Review | 13. Add User
10. Update Review | 14. Update User
11. Search Review | 15. Search User
12. Delete Review | 16. Delete User

=====
17. Exit

Choose an option: 5
Enter Menu ID: 5
Enter Menu Name: 보리삼치
Enter Restaurant ID: 4
Enter Price: 24000
Menu added successfully.

```

- 관리자가 메뉴를 추가할 때에는 순서대로 Menu ID, Menu Name, Restaurant ID, price를 입력해야한다.

메뉴 추가 전)

```

mysql> SELECT * FROM DB2024_Menu WHERE res_id=4;
+-----+-----+-----+-----+
| menu_id | menu_name | res_id | price |
+-----+-----+-----+-----+
1	전복 관자 파스타	4	23000
2	홍새우 파스타	4	23000
3	흑돼지 핑크안심	4	17000
4	E.C.C 샐러드	4	16000
+-----+-----+-----+-----+
4 rows in set (0.00 sec)

```

메뉴 추가 후)

| menu_id | menu_name | res_id | price |
|---------|-----------|--------|-------|
| 1       | 전복 관자 파스타 | 4      | 23000 |
| 2       | 홍새우 파스타   | 4      | 23000 |
| 3       | 흑돼지 핑크안심  | 4      | 17000 |
| 4       | E.C.C 샐러드 | 4      | 16000 |
| 5       | 보리삼치      | 4      | 24000 |

5 rows in set (0.00 sec)

### ▼ 3. 식당 추가

```
===== Restaurant ===== ====== Menu =====

1. Add Restaurant | 5. Add Menu
2. Update Restaurant | 6. Update Menu
3. Search Restaurant | 7. Search Menu
4. Delete Restaurant | 8. Delete Menu

===== Review ===== ====== User =====

9. Add Review | 13. Add User
10. Update Review | 14. Update User
11. Search Review | 15. Search User
12. Delete Review | 16. Delete User

=====
17. Exit

Choose an option: 1
Enter Restaurant Name: 사장님돈까스 이화여대점
Enter Restaurant ID: 32
Enter Phone Number: 0507-1460-2280
Enter Address: 서울 서대문구 이화여대7길 11 1층
Enter Operating Hours: 월~금 10:00~20:50
Enter Break Time:
Enter Rating (choose from 1 to 5): 4
Enter Cuisine Type: 일식
Enter Location: 정문
Restaurant added successfully.
```

- 관리자가 신규 식당을 등록할 때에는 순서대로 Restaurant name, Restaurant ID, phone number, address, operating hours, break time, rating, cuisine type, location을 입력해야한다.

식당 추가 전)

| res_name         | res_id              | phone_num                | address                        | operating_hours | break_time | rating | cuisine_type | location |
|------------------|---------------------|--------------------------|--------------------------------|-----------------|------------|--------|--------------|----------|
| 모리수식당            | 1   070-4154-2800   | 서울 서대문구 이화여대길 24 2층      | 월~금 11:00~20:30, 토 11:30~20:30 | 월~토 15:00~17:00 | 4.8   일식   | 경문     |              |          |
| 남한국식당            | 2   02-312-1238     | 서울특별시 서대문구 이화여대길 6 1층    | 월~금 11:00~20:00                | 월~금 15:00~17:00 | NULL   일식  | 경문     |              |          |
| 워즈카운             | 3   02-313-2198     | 서울 서대문구 이화여대길 20 1층      | 월~토 10:00~21:00                | NULL   일식       | 3.0   베이커리 | 경문     |              |          |
| 심플카페스            | 4   010-2583-3196   | 서울 서대문구 이화여대길 24 2층      | 월~토 11:30~21:30                | 월~토 15:00~17:00 | 4.0   양식   | 경문     |              |          |
| 조식카페스            | 5   02-365-5679     | 서울 서대문구 이화여대길 19 1층      | 월~금 11:00~19:30                | 월~금 15:00~17:00 | 4.0   샐러드  | 경문     |              |          |
| 조식카페스            | 6   02-3401-7991    | 서울 서대문구 이화여대길 52-1 1층    | 월~금 11:00~20:00                | 월~금 15:00~17:00 | 3.0   일식   | 경문     |              |          |
| 우미카페             | 7   NULL            | 서울특별시 서대문구 신촌로 17 1층     | 월~일 11:00~21:00                | 월~일 15:00~17:00 | 3.0   일식   | 경문     |              |          |
| 아온카페             | 8   02-364-1380     | 서울특별시 서대문구 신촌로 17 1층     | 월~금 11:00~21:00                | 월~금 15:00~17:00 | 5.0   도시락  | 경문     |              |          |
| 진돈수미리            | 9   010-4726-7686   | 서울 서대문구 신촌로 16 신촌 거리 104 | 월~토 17:00~20:20                | NULL   일식       | 4.0   일식   | 경문     |              |          |
| 어반카페스            | 10   02-4407-4917   | 서울 서대문구 이화여대길 28 1층      | 월~금 10:30~20:00                | 월~토 15:00~17:00 | 4.0   일식   | 경문     |              |          |
| 오스카카페            | 11   010-4711-3917  | 서울 서대문구 이화여대길 58-1 1층    | 월~금 11:00~20:00                | 월~금 15:00~17:00 | 3.0   일식   | 경문     |              |          |
| 까비식당             | 12   070-7779-8899  | 서울 서대문구 이화여대길 24 1층      | 월~금 11:00~21:00                | 월~금 15:00~17:00 | 4.0   아시안  | 경문     |              |          |
| 아민 이화            | 13   02-363-8113    | 서울 서대문구 이화여대길 52-31 1층   | 월~일 11:30~21:30                | 월~일 15:00~17:00 | 4.0   양식   | 경문     |              |          |
| 유소카페             | 14   070-8224-7956  | 서울 서대문구 이화여대길 4 1층       | 월~금 11:00~20:00                | 월~금 15:00~17:00 | 4.0   일식   | 경문     |              |          |
| 조식카페트날랄국수 이대점    | 15   02-365-1757    | 서울특별시 서대문구 이화여대길 29 101호 | 월~일 9:00~18:00                 | 월~일 10:30~20:00 | 2.0   도시락  | 경문     |              |          |
| 마더닭 카운           | 16   070-1413-4268  | 서울 서대문구 이화여대길 24 2층      | 월~금 10:30~20:00                | 월~금 15:00~16:00 | 4.0   양식   | 경문     |              |          |
| 티아리 카페스타         | 17   0507-1413-4268 | 서울 서대문구 이화여대길 24 2층      | 월~금 10:30~20:00                | 월~금 15:00~17:00 | 4.0   양식   | 경문     |              |          |
| 스텐情人节 카페         | 18   02-365-6353    | 서울 서대문구 연대동문길 49 1층      | 월~금 10:30~20:00                | 월~금 15:00~17:00 | 3.0   일식   | 경문     |              |          |
| 이로비 이성장          | 19   02-365-1245    | 서울 서대문구 연대동문길 27-2 2층    | 월~금 10:30~20:00                | 월~금 15:00~17:00 | 3.0   한식   | 경문     |              |          |
| 화교중국집            | 20   02-365-1245    | 서울 서대문구 연대동문길 27-2 2층    | 월~금 10:30~20:00                | 월~금 15:00~17:00 | 3.0   한식   | 경문     |              |          |
| 화교중국집            | 21   02-723-4884    | 서울 서대문구 성산로 52 1층        | 월~일 10:00~21:00                | 월~일 15:00~17:00 | 3.0   양식   | 후문     |              |          |
| 식탁 하노슬점          | 22   02-362-8777    | 서울 서대문구 성산로 52 18노슬빌딩    | 월~일 11:00~20:00                | 월~일 15:00~16:30 | 1.0   양식   | 후문     |              |          |
| 풀밭               | 23   02-362-9833    | 서울 서대문구 연대동문길 11 2층      | 월~일 11:00~22:00                | 월~일 15:00~17:00 | 3.0   한식   | 경문     |              |          |
| 주부 이대점           | 24   02-362-9333    | 서울 서대문구 연대동문길 52-1 1층    | 월~일 11:00~22:00                | 월~일 15:00~17:00 | 3.0   일식   | 경문     |              |          |
| 의원상              | 25   NULL           | 서울 서대문구 이화여대길 72-5 1층    | 월~일 10:30~21:30                | 월~일 15:00~17:00 | 3.0   중식   | 경문     |              |          |
| 화코리마라탕           | 26   02-313-8151    | 서울 서대문구 이화여대길 76 1층      | 월~일 10:00~22:00                | 월~일 15:00~17:00 | 5.0   일식   | 경문     |              |          |
| 마더닭카페 이글         | 27   070-7758-3838  | 서울 서대문구 이화여대길 5 지성 1층    | 월~금 8:30~16:30, 토 10:00~16:30  | 월~금 10:30~20:00 | 2.0   중식   | 경문     |              |          |
| 이화여대점            | 28   070-7758-3838  | 서울 서대문구 이화여대길 5 지성 1층    | 월~금 8:30~16:30, 토 10:00~16:30  | 월~금 10:30~20:00 | 2.0   중식   | 경문     |              |          |
| 한국상상문화가 이화여자대학교점 | 29   0507-1344-6031 | 서울 서대문구 이화여대길 52-1 지하 1층 | 월~일 11:00~22:00                | 월~일 15:00~17:00 | 4.0   일식   | 경문     |              |          |
| 등록서비스카페 카페수 신촌점  | 30   0507-1398-7858 | 서울 서대문구 신촌로 43 1층        | 월~일 11:00~22:00                | 월~일 15:00~17:00 | 5.0   샌드위치 | 경문     |              |          |
| 슬로우轲리 이대점        | 31   0507-1391-7188 | 서울 서대문구 이화여대길 78 가동 1층   | 월~일 11:00~21:00                | 월~일 15:00~17:00 | 4.0   샐러드  | 경문     |              |          |

식당 추가 후)

| res_name         | res_id              | phone_num                | address                        | operating_hours | break_time | rating | cuisine_type | location |
|------------------|---------------------|--------------------------|--------------------------------|-----------------|------------|--------|--------------|----------|
| 모리수식당            | 1   070-4154-2800   | 서울 서대문구 이화여대길 24 2층      | 월~금 11:00~20:30, 토 11:30~20:30 | 월~토 15:00~17:00 | 4.8   일식   | 경문     |              |          |
| 남한국식당            | 2   02-312-1238     | 서울 서대문구 이화여대길 6 1층       | 월~금 11:00~20:00                | 월~금 15:00~17:00 | NULL   일식  | 경문     |              |          |
| 워즈카운             | 3   02-313-2198     | 서울 서대문구 이화여대길 20 1층      | 월~토 10:00~21:00                | NULL   일식       | 3.0   베이커리 | 경문     |              |          |
| 심플카페스            | 4   010-2583-3196   | 서울 서대문구 이화여대길 24 2층      | 월~토 11:30~21:30                | 월~토 15:00~17:00 | 4.0   샐러드  | 경문     |              |          |
| 조식카페스            | 5   02-365-5679     | 서울 서대문구 이화여대길 19 1층      | 월~금 11:00~19:30                | 월~금 15:00~17:00 | 3.0   일식   | 경문     |              |          |
| 조식카페스            | 6   02-3401-7991    | 서울 서대문구 이화여대길 52-1 1층    | 월~금 11:00~20:00                | 월~금 15:00~17:00 | 4.0   일식   | 경문     |              |          |
| 우미카페             | 7   NULL            | 서울 서대문구 이화여대길 72-5 1층    | 월~일 11:00~21:00                | 월~일 15:00~17:00 | 5.0   도시락  | 경문     |              |          |
| 아온카페             | 8   02-364-1380     | 서울특별시 서대문구 신촌로 17 1층     | 월~금 11:00~21:00                | 월~금 15:00~17:00 | NULL   일식  | 경문     |              |          |
| 진돈수미리            | 9   010-4726-7686   | 서울 서대문구 신촌로 16 신촌 거리 104 | 월~토 17:00~20:20                | NULL   일식       | 4.0   일식   | 경문     |              |          |
| 어반카페스            | 10   02-4407-4917   | 서울 서대문구 이화여대길 28 1층      | 월~금 10:30~20:00                | 월~토 15:00~17:00 | 4.0   일식   | 경문     |              |          |
| 오스카카페            | 11   010-4711-3917  | 서울 서대문구 이화여대길 58-1 1층    | 월~금 11:00~20:00                | 월~금 15:00~17:00 | 3.0   일식   | 경문     |              |          |
| 까비식당             | 12   070-7779-8899  | 서울 서대문구 이화여대길 24 1층      | 월~금 11:00~21:00                | 월~금 15:00~17:00 | 4.0   아시안  | 경문     |              |          |
| 아민 이화            | 13   02-363-8113    | 서울 서대문구 이화여대길 52-31 1층   | 월~일 11:30~21:30                | 월~일 15:00~17:00 | 4.0   양식   | 경문     |              |          |
| 유소카페             | 14   070-8224-7956  | 서울 서대문구 이화여대길 4 1층       | 월~금 11:00~20:00                | 월~금 15:00~17:00 | 4.0   일식   | 경문     |              |          |
| 조식카페트날랄국수 이대점    | 15   02-365-1757    | 서울특별시 서대문구 이화여대길 29 101호 | 월~일 9:00~18:00                 | 월~일 10:30~20:00 | 2.0   도시락  | 경문     |              |          |
| 마더닭 카운           | 16   070-1413-4268  | 서울 서대문구 이화여대길 24 2층      | 월~일 11:00~21:00                | 월~일 15:00~16:00 | 4.0   양식   | 경문     |              |          |
| 티아리 카페스타         | 17   0507-1413-4268 | 서울 서대문구 이화여대길 24 2층      | 월~일 11:00~21:00                | 월~일 15:00~17:00 | 4.0   양식   | 경문     |              |          |
| 스텐情人节 카페         | 18   02-365-6353    | 서울 서대문구 연대동문길 49 1층      | 월~일 11:00~21:00                | 월~일 15:00~17:00 | 3.0   일식   | 경문     |              |          |
| 이로비 이성장          | 19   02-365-1245    | 서울 서대문구 연대동문길 27-2 2층    | 월~일 11:00~21:00                | 월~일 15:00~17:00 | 3.0   한식   | 경문     |              |          |
| 화교중국집            | 20   02-365-1245    | 서울 서대문구 연대동문길 27-2 2층    | 월~일 11:00~21:00                | 월~일 15:00~17:00 | 3.0   한식   | 경문     |              |          |
| 화교중국집            | 21   02-723-4884    | 서울 서대문구 성산로 52 1층        | 월~일 10:00~21:00                | 월~일 15:00~17:00 | 3.0   양식   | 후문     |              |          |
| 식탁 하노슬점          | 22   02-362-8777    | 서울 서대문구 성산로 52 18노슬빌딩    | 월~일 10:00~21:00                | 월~일 15:00~16:30 | 1.0   양식   | 후문     |              |          |
| 풀밭               | 23   02-362-9833    | 서울 서대문구 연대동문길 11 2층      | 월~일 11:00~22:00                | 월~일 15:00~16:30 | 3.0   한식   | 경문     |              |          |
| 주부 이대점           | 24   02-362-9333    | 서울 서대문구 연대동문길 52-1 1층    | 월~일 11:00~22:00                | 월~일 15:00~17:00 | 3.0   양식   | 경문     |              |          |
| 의원상              | 25   NULL           | 서울 서대문구 이화여대길 72-5 1층    | 월~일 10:00~21:30                | 월~일 15:00~17:00 | 3.0   중식   | 경문     |              |          |
| 화코리마라탕           | 26   02-313-8151    | 서울 서대문구 이화여대길 76 1층      | 월~일 10:00~22:00                | 월~일 15:00~17:00 | 3.0   중식   | 경문     |              |          |
| 마더닭카페 이글         | 27   070-7758-3838  | 서울 서대문구 이화여대길 5 지성 1층    | 월~금 8:30~16:30, 토 10:00~16:30  | 월~금 10:30~20:00 | 2.0   중식   | 경문     |              |          |
| 이화여대점            | 28   070-7758-3838  | 서울 서대문구 이화여대길 5 지성 1층    | 월~금 8:30~16:30, 토 10:00~16:30  | 월~금 10:30~20:00 | 2.0   중식   | 경문     |              |          |
| 한국상상문화가 이화여자대학교점 | 29   0507-1344-6031 | 서울 서대문구 이화여대길 52-1 지하 1층 | 월~일 11:00~22:00                | 월~일 15:00~17:00 | 4.0   일식   | 경문     |              |          |
| 등록서비스카페 카페수 신촌점  | 30   0507-1398-7858 | 서울 서대문구 신촌로 43 1층        | 월~일 11:00~22:00                | 월~일 15:00~17:00 | 5.0   샌드위치 | 경문     |              |          |
| 슬로우轲리 이대점        | 31   0507-1391-7188 | 서울 서대문구 이화여대길 78 가동 1층   | 월~일 11:00~21:00                | 월~일 15:00~17:00 | 4.0   일식   | 경문     |              |          |
| 사장님돈까스 이화여대점     | 32   0507-1460-2280 | 서울 서대문구 이화여대길 11 1층      | 월~금 10:00~20:50                |                 | 4.0   일식   | 경문     |              |          |

#### ▼ 4. 리뷰 추가

```
Welcome to E-MATEASY, astralfinance!
=====
1. Search for restaurants
2. Get Random Restaurant
3. Posting a new review
4. My Page
5. Logout

Choose an option: 3
Enter User ID: astralfinance
Enter Restaurant ID: 1
Enter Menu ID: 2
Enter Rating (1-5): 5
Enter review comment: 가지가 너무 부드럽고 소스가 적당히 베어들어서 맛있어요ㅠㅠ !
Review added successfully.
Restaurant rating updated successfully.
```

- 사용자가 신규 리뷰를 등록할 때에는 순서대로 user ID, restaurant ID, menu ID, rating, review comment를 입력해야한다.

리뷰 추가 전)

```
mysql> SELECT * FROM DB2024_Review WHERE user_id='astralfinance';
+-----+-----+-----+
| review_id | user_id | rating | review_content |
+-----+-----+-----+
5	astralfinance	5	간장새우계란밥이 진짜 최고에요.. 눈물 흘리면서 먹었어요
9	astralfinance	3	그냥 참봉비르에요. 무난합니다.
16	astralfinance	4	등심돈까스 무난하게 맛있어요~
19	astralfinance	5	봇가게 진짜 맛있어요... 꼭 드셔보세요 다들!
30	astralfinance	5	버섯 샴브세트 삼겹살만큼 맛있어요!
36	astralfinance	5	타코 비슷한 맛도 나는 것 같아요. 나쁘지않아요!
45	astralfinance	3	아무래도 소불고기가 가격대가 좀 있어요.. 맛있긴한데 아쉽ㅠㅠ
57	astralfinance	4	꾸덕하니 맛있네요!!
62	astralfinance	3	양념이 너무 달아요ㅠㅠ
+-----+-----+-----+
9 rows in set (0.00 sec)
```

리뷰 추가 후)

```
mysql> SELECT * FROM DB2024_Review WHERE user_id='astralfinance';
+-----+-----+-----+
| review_id | user_id | rating | review_content |
+-----+-----+-----+
5	astralfinance	5	간장새우계란밥이 진짜 최고에요.. 눈물 흘리면서 먹었어요
9	astralfinance	3	그냥 참봉비르에요. 무난합니다.
16	astralfinance	4	등심돈까스 무난하게 맛있어요~
19	astralfinance	5	봇가게 진짜 맛있어요... 꼭 드셔보세요 다들!
30	astralfinance	5	버섯 샴브세트 삼겹살만큼 맛있어요!
36	astralfinance	5	타코 비슷한 맛도 나는 것 같아요. 나쁘지않아요!
45	astralfinance	3	아무래도 소불고기가 가격대가 좀 있어요.. 맛있긴한데 아쉽ㅠㅠ
57	astralfinance	4	꾸덕하니 맛있네요!!
62	astralfinance	3	양념이 너무 달아요ㅠㅠ
63	astralfinance	5	가지가 너무 부드럽고 소스가 적당히 베어들어서 맛있어요ㅠㅠ!!
+-----+-----+-----+
10 rows in set (0.00 sec)
```

▼ (15) 데이터베이스에 갱신(update)을 하기 위한 인터페이스와 쿼리를 가지고 있어야 한다.

▼ 1. 회원정보 수정

```
===== My Page =====

1. My information
2. Update My information
3. Search for Other Users
4. Delete My Account
5. Exit

=====
Choose an option: 2

===== Update My information =====

New Password (Press enter to skip): danchoo
New Name (Press enter to skip):
New Student ID (Press enter to skip):
New Email (Press enter to skip):
New Location (Press enter to skip): 정문
Update successful!
```

- 사용자가 회원정보를 수정할 때에는 user\_id를 제외한 나머지 정보들 (password, name, student ID, email, location) 에 대해서 선택적으로 새로운 정보를 입력하면 해당 정보만 update 된다.

회원정보 수정 전)

```
mysql> SELECT * FROM DB2024_User;
+-----+-----+-----+-----+-----+-----+
| user_id | user_pw | name   | student_id | email           | location |
+-----+-----+-----+-----+-----+-----+
| astralfinance | astralfinance | 한사랑 | 2271064 | astralfinance@ewhain.net | 후문
| cannes7 | cannes7 | 고은서 | 2122004 | cannes7@ewhain.net | 정문
| chacha091 | chacha091 | 차현주 | 2276321 | chacha09@ewhain.net | 정문
| meanwest | meanwest | 김민서 | 2276046 | meanwestk@gmail.com | 후문
| s2eojeong | s2eojeong | 조서정 | 2276305 | s2eojeong@gmail.com | 후문
+-----+-----+-----+-----+-----+
5 rows in set (0.00 sec)
```

회원정보 수정 후)

```
mysql> SELECT * FROM DB2024_User;
+-----+-----+-----+-----+-----+-----+
| user_id | user_pw | name   | student_id | email           | location |
+-----+-----+-----+-----+-----+-----+
| astralfinance | astralfinance | 한사랑 | 2271064 | astralfinance@ewhain.net | 후문
| cannes7 | cannes7 | 고은서 | 2122004 | cannes7@ewhain.net | 정문
| chacha091 | chacha091 | 차현주 | 2276321 | chacha09@ewhain.net | 정문
| meanwest | meanwest | 김민서 | 2276046 | meanwestk@gmail.com | 후문
| s2eojeong | danchoo | 조서정 | 2276305 | s2eojeong@gmail.com | 정문
+-----+-----+-----+-----+-----+
5 rows in set (0.00 sec)
```

## ▼ 2. 메뉴 수정

| Choose an option: 6 |                 |
|---------------------|-----------------|
| Restaurant ID       | Restaurant Name |
| 1                   | 모미지식당           |
| 2                   | 낭만식탁            |
| 3                   | 원즈오운            |
| 4                   | 심플리스트           |
| 5                   | 초식곳간            |
| 6                   | 유아케도쿄           |
| 7                   | 우미마루            |
| 8                   | 아콘스틀            |
| 9                   | 진돈부리            |
| 10                  | 어비웃샤브           |
| 11                  | 소오밥집            |
| 12                  | 까이식당            |
| 13                  | 아민 이화           |
| 14                  | 유소바             |
| 15                  | 포히엔베트남쌀국수 이대점   |
| 16                  | 마더락 신촌점         |
| 17                  | 티아라 파스타         |
| 18                  | 스텐바이키친          |
| 19                  | 이로운 밥상          |
| 20                  | 화가와요리사 이대후문점    |
| 21                  | 헬리우드            |
| 22                  | 식탁 하늬솔점         |
| 23                  | 불밥              |
| 24                  | 수라 이대점          |
| 25                  | 의원상             |
| 26                  | 화리라마라탕          |
| 27                  | 마더린더베이글         |
| 28                  | 이화성             |
| 29                  | 청년밥상문간 이화여자대학교점 |
| 30                  | 등촌사브칼국수 신촌점     |
| 31                  | 슬로우캘리 이대점       |

```

Enter Restaurant ID: 4
-----
Menu ID      Menu Name
-----
1           전복 관자 파스타
2           흥새우 파스타
3           흑돼지 핑크안심
4           E.C.C 샐러드
-----
Enter Menu ID: 2
Enter New Menu Name (Press enter to skip):
Enter New Price (Press enter to skip): 25000
Menu updated successfully.

```

- 관리자가 메뉴 수정을 할 경우에는 res\_id와 restaurant name을 리스트로 보여준 후, 수정을 원하는 메뉴가 있는 식당의 res\_id를 입력하면, 해당 식당에 존재하는 모든 메뉴들의 리스트들을 보여준다. 이후 수정을 원하는 menu\_id를 입력한 후 수정을 원하는 항목에 선택적으로 새로운 값을 입력하면 해당 값으로 메뉴가 update 된다.

(메뉴 수정 전)

```

mysql> SELECT * FROM DB2024_Menu WHERE res_id=4;
+-----+-----+-----+-----+
| menu_id | menu_name          | res_id | price |
+-----+-----+-----+-----+
1	전복 관자 파스타	4	23000
2	흥새우 파스타	4	23000
3	흑돼지 핑크안심	4	17000
4	E.C.C 샐러드	4	16000
+-----+-----+-----+-----+
4 rows in set (0.00 sec)

```

(메뉴 수정 후)

```

mysql> SELECT * FROM DB2024_Menu WHERE res_id=4;
+-----+-----+-----+-----+
| menu_id | menu_name          | res_id | price |
+-----+-----+-----+-----+
1	전복 관자 파스타	4	23000
2	흥새우 파스타	4	25000
3	흑돼지 핑크안심	4	17000
4	E.C.C 샐러드	4	16000
+-----+-----+-----+-----+
4 rows in set (0.00 sec)

```

### ▼ 3. 식당 수정

| Choose an option: 6 |                 |
|---------------------|-----------------|
| Restaurant ID       | Restaurant Name |
| 1                   | 모미지식당           |
| 2                   | 낭만식탁            |
| 3                   | 원즈오운            |
| 4                   | 심플리스트           |
| 5                   | 초식곳간            |
| 6                   | 유아케도쿄           |
| 7                   | 우미마루            |
| 8                   | 아콘스톨            |
| 9                   | 진돈부리            |
| 10                  | 어비웃샤브           |
| 11                  | 소오밥집            |
| 12                  | 끼이식당            |
| 13                  | 아민 이화           |
| 14                  | 유소바             |
| 15                  | 포히엔베트남쌀국수 이대점   |
| 16                  | 마더락 신촌점         |
| 17                  | 티아라 파스타         |
| 18                  | 스탠바이키친          |
| 19                  | 이로운 밥상          |
| 20                  | 화가와오리사 이대후문점    |
| 21                  | 헬리우드            |
| 22                  | 식탁 하늬솔점         |
| 23                  | 불밥              |
| 24                  | 수라 이대점          |
| 25                  | 의원상             |
| 26                  | 화라리마리탕          |
| 27                  | 마더린더베이글         |
| 28                  | 이화성             |
| 29                  | 청년밥상문간 이화여자대학교점 |
| 30                  | 등촌샤브칼국수 신촌점     |
| 31                  | 슬로우캘리 이대점       |

```
Enter Restaurant ID: 4
Enter New Restaurant Name (Press enter to skip): 복잡리스트
Enter New Phone Number (Press enter to skip): 010-1234-5678
Enter New Address (Press enter to skip):
Enter New Operating Hours (Press enter to skip):
Enter New Break Time (Press enter to skip):
Enter New Rating (Press enter to skip):
Enter New Cuisine Type (Press enter to skip):
Enter New Location (Press enter to skip):
Restaurant updated successfully.
```

- 관리자가 식당 정보를 수정 할 경우에는, res\_id와 restaurant name을 리스트로 보여준 후, 수정을 원하는 식당의 res\_id를 입력한 후 수정을 원하는 항목에 선택적으로 새로운 값을 입력하면 해당 값으로 식당 정보가 update 된다.

#### 식당 수정 전)

```
mysql> SELECT * FROM DB2024_Restaurant WHERE res_id=4;
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| res_name | res_id | phone_num | address          | operating_hours | break_time    | rating | cuisine_type | location |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 심플리스트 | 4 | 010-2583-3190 | 서울 서대문구 이화여대길 24 2층 | 월~토 11:30~21:30 | 월~토 15:00~17:00 | 4.0 | 양식 | 경문 |
+-----+-----+-----+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)
```

#### 식당 수정 후)

```
mysql> SELECT * FROM DB2024_Restaurant WHERE res_id=4;
+-----+-----+-----+-----+-----+-----+-----+-----+
| res_name | res_id | phone_num | address | operating_hours | break_time | rating | cuisine_type | location |
+-----+-----+-----+-----+-----+-----+-----+-----+
| 불갈리스트 | 4 | 010-1234-5678 | 서울 서대문구 이화여대길 24 2층 | 월~토 11:30~21:30 | 월~토 15:00~17:00 | 4.0 | 양식 | 경문 |
+-----+-----+-----+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)
```

#### ▼ 4. 리뷰 수정

```
===== Restaurant ===== ===== Menu =====
1. Add Restaurant | 5. Add Menu
2. Update Restaurant | 6. Update Menu
3. Search Restaurant | 7. Search Menu
4. Delete Restaurant | 8. Delete Menu

===== Review ===== ===== User =====
9. Add Review | 13. Add User
10. Update Review | 14. Update User
11. Search Review | 15. Search User
12. Delete Review | 16. Delete User

=====
17. Exit

Choose an option: 10
Enter review ID to update: 10
Enter rating (1-5) to update: 5
Enter review content to update: 저는 홍새우 파스타 바다 향이 나서 좋았는데 향이 좀 강해서 호불호 갈릴 수도 있어요!
Review updated successfully.
```

- 관리자가 리뷰 수정을 할 경우에는, 수정을 원하는 review\_id를 입력한 후 수정을 원하는 항목에 선택적으로 새로운 값을 입력하면 해당 값으로 리뷰가 update 된다.

리뷰 수정 전)

```
mysql> SELECT * FROM DB2024_Review WHERE review_id=10;
+-----+-----+-----+-----+
| review_id | user_id | rating | review_content |
+-----+-----+-----+-----+
| 10 | meanwest | 5 | 홍새우 파스타 바다 향 솔솔나고 맛있어요! |
+-----+-----+-----+-----+
1 row in set (0.00 sec)
```

리뷰 수정 후)

```
mysql> SELECT * FROM DB2024_Review WHERE review_id=10;
+-----+-----+-----+-----+
| review_id | user_id | rating | review_content |
+-----+-----+-----+-----+
| 10 | meanwest | 5 | 저는 홍새우 파스타 바다 향이 나서 좋았는데 향이 좀 강해서 호불호 갈릴 수도 있어요! |
+-----+-----+-----+-----+
1 row in set (0.00 sec)
```

#### ▼ (16) 데이터베이스에 삭제(delete)를 하기 위한 인터페이스와 쿼리를 가지고 있어야 한다.

##### ▼ 1. 회원정보 삭제

```

===== My Page =====

1. My information
2. Update My information
3. Search for Other Users
4. Delete My Account
5. Exit

=====
Choose an option: 4

Are you sure you want to delete your account?
All data related to your will be deleted.

Enter 'y' if you want to delete, or 'n': y
Please enter your PASSWORD: s2eojeong
Membership withdrawal completed.

```

- 사용자가 탈퇴를 하는 경우에는 'Are you sure you want to delete your account?' 와 'All data related to you will be deleted' 라는 경고 문구를 띄운 후, 사용자가 'y'로 답하는 경우 password를 입력받고, 일치할 경우에만 삭제한다.

(회원정보 삭제 전)

```

mysql> SELECT * FROM DB2024_User;
+-----+-----+-----+-----+-----+-----+
| user_id | user_pw | name | student_id | email | location |
+-----+-----+-----+-----+-----+-----+
astralfinance	astralfinance	한사랑	2271064	astralfinance@ewhain.net	후문
cannes7	cannes7	고은서	2122004	cannes7@ewhain.net	정문
chacha091	chacha091	차현주	2276321	chacha09@ewhain.net	정문
meanwest	meanwest	김민서	2276046	meanwestk@gmail.com	후문
s2eojeong	s2eojeong	조서정	2276305	s2eojeong@gmail.com	후문
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.00 sec)

```

(회원정보 삭제 후)

```

mysql> SELECT * FROM DB2024_User;
+-----+-----+-----+-----+-----+-----+
| user_id | user_pw | name | student_id | email | location |
+-----+-----+-----+-----+-----+-----+
astralfinance	astralfinance	한사랑	2271064	astralfinance@ewhain.net	후문
cannes7	cannes7	고은서	2122004	cannes7@ewhain.net	정문
chacha091	chacha091	차현주	2276321	chacha09@ewhain.net	정문
meanwest	meanwest	김민서	2276046	meanwestk@gmail.com	후문
+-----+-----+-----+-----+-----+-----+
4 rows in set (0.00 sec)

```

## ▼ 2. 메뉴 삭제

| Choose an option: 8 |                 |
|---------------------|-----------------|
| Restaurant ID       | Restaurant Name |
| 1                   | 모미자식당           |
| 2                   | 낭만식탁            |
| 3                   | 원즈오운            |
| 4                   | 복잡리스트           |
| 5                   | 초식곳간            |
| 6                   | 유아케도쿄           |
| 7                   | 우미마루            |
| 8                   | 아콘스틀            |
| 9                   | 진돈부리            |
| 10                  | 어바웃샤브           |
| 11                  | 소오밥집            |
| 12                  | 까이식당            |
| 13                  | 아민 이화           |
| 14                  | 유소바             |
| 15                  | 포히엔베트남쌀국수 이대점   |
| 16                  | 마더락 신촌점         |
| 17                  | 티아라 파스타         |
| 18                  | 스탠바이키친          |
| 19                  | 이로운 밥상          |
| 20                  | 화가와요리사 이대후문점    |
| 21                  | 헬리우드            |
| 22                  | 식탁 하늬솔점         |
| 23                  | 불밥              |
| 24                  | 수라 이대점          |
| 25                  | 의원상             |
| 26                  | 화라라마라탕          |
| 27                  | 마더린더베이글         |
| 28                  | 이화성             |
| 29                  | 청년밥상문간 이화여자대학교점 |
| 30                  | 등촌샤브칼국수 신촌점     |
| 31                  | 슬로우캘리 이대점       |

| Enter Restaurant ID: 6 |           |
|------------------------|-----------|
| Menu ID                | Menu Name |
| 1                      | 등심돈까스     |
| 2                      | 안심돈까스     |
| 3                      | 치즈돈까스     |

| Enter Menu ID: 3           |  |
|----------------------------|--|
| Menu deleted successfully. |  |

- 관리자가 메뉴 삭제를 할 경우에는, res\_id와 restaurant name을 리스트로 보여준 후, 삭제를 원하는 메뉴가 있는 식당의 res\_id를 입력하면, 해당 식당에 존재하는 모든 메뉴들의 리스트들을 보여준다. 이후 삭제를 원하는 menu\_id를 입력하면 해당 menu가 삭제된다.

메뉴 삭제 전)

```

mysql> SELECT * FROM DB2024_Menu WHERE res_id=6;
+-----+-----+-----+
| menu_id | menu_name      | res_id | price |
+-----+-----+-----+
1	등심돈까스	6	9500
2	안심돈까스	6	11000
3	치즈돈까스	6	11000
+-----+-----+-----+
3 rows in set (0.01 sec)

```

메뉴 삭제 후)

```

mysql> SELECT * FROM DB2024_Menu WHERE res_id=6;
+-----+-----+-----+
| menu_id | menu_name      | res_id | price |
+-----+-----+-----+
| 1       | 등심돈까스      | 6      | 9500  |
| 2       | 안심돈까스      | 6      | 11000 |
+-----+-----+-----+
2 rows in set (0.00 sec)

```

```

mysql> SELECT * FROM DB2024_User;
+-----+-----+-----+-----+-----+-----+
| user_id   | user_pw     | name    | student_id | email           | location |
+-----+-----+-----+-----+-----+-----+
astralfinance	astralfinance	한사랑	2271064	astralfinance@ewhain.net	후문
cannes7	cannes7	고은서	2122004	cannes7@ewhain.net	정문
chacha091	chacha091	차현주	2276321	chacha09@ewhain.net	정문
meanwest	meanwest	김민서	2276046	meanwestk@gmail.com	후문
+-----+-----+-----+-----+-----+-----+
4 rows in set (0.00 sec)

```

### ▼ 3. 식당 삭제

| Choose an option: 4 |                 |
|---------------------|-----------------|
| Restaurant ID       | Restaurant Name |
| 1                   | 모미지식당           |
| 2                   | 남만식탁            |
| 3                   | 원즈오운            |
| 4                   | 복잡리스트           |
| 5                   | 초식곳간            |
| 6                   | 유아케도쿄           |
| 7                   | 우미마루            |
| 8                   | 아콘스틀            |
| 9                   | 진돈부리            |
| 10                  | 어비웃샤브           |
| 11                  | 소오밥집            |
| 12                  | 까이식당            |
| 13                  | 아민 이화           |
| 14                  | 유소바             |
| 15                  | 포히엔베트남쌀국수 이대점   |
| 16                  | 마더락 신촌점         |
| 17                  | 티이라 파스타         |
| 18                  | 스탠바이키친          |
| 19                  | 이로운 밥상          |
| 20                  | 화가와요리사 이대후문점    |
| 21                  | 헐리우드            |
| 22                  | 식탁 하늬솔점         |
| 23                  | 불밥              |
| 24                  | 수라 이대점          |
| 25                  | 의원상             |
| 26                  | 화리라미마리탕         |
| 27                  | 마더린베이글          |
| 28                  | 이화성             |
| 29                  | 청년밥상문간 이화여자대학교점 |
| 30                  | 등촌샤브칼국수 신촌점     |
| 31                  | 슬로우캘리 이대점       |

Enter Restaurant ID: 9  
Restaurant deleted successfully.

- 관리자가 식당 삭제를 할 경우에는, res\_id와 restaurant name을 리스트로 보여준 후, 삭제를 원하는 식당의 res\_id를 입력하면 해당 식당이 삭제된다.

### 식당 삭제 전)

| res_name        | res_id | phone_num      | address                     | operating_hours                | break_time        | rating | cuisine_type | location |
|-----------------|--------|----------------|-----------------------------|--------------------------------|-------------------|--------|--------------|----------|
| 모미지식당           | 1      | 070-4154-2000  | 서울 서대문구 이화여대길 79 2층         | 월~금 11:00~20:30, 토 11:30~20:30 | 월~일 15:00~17:00   | 4.0    | 일식           | 경문       |
| 남만식탁            | 2      | 02-313-1111    | 서울 서대문구 이화여대길 5 1층          | 월~토 10:00~21:00                | NNULL             | 3.0    | 베이커리         | 경문       |
| 원즈오운            | 3      | 02-313-3130    | 서울 특별시 서대문구 이화여대길 28 1층     | 월~토 11:30~21:30                | 월~토 15:00~17:00   | 4.0    | 양식           | 경문       |
| 복잡리스트           | 4      | 010-2583-3190  | 서울 서대문구 이화여대길 24 2층         | 월~금 11:00~19:30                | NNULL             | 4.0    | 샐러드          | 경문       |
| 초식곳간            | 5      | 02-365-5679    | 서울 서대문구 이화여대길 19 1층         | 월~금 11:00~21:00                | NNULL             | 3.0    | 한식           | 경문       |
| 유아케도쿄           | 6      | 02-6401-7991   | 서울 서대문구 이화여대길 18 1층         | 월~금 11:00~21:00                | NNULL             | 3.0    | 일식           | 경문       |
| 어비웃샤브           | 7      | 02-364-1300    | 서울 특별시 서대문구 이화여대길 1 1층      | 월~일 11:00~21:00                | 월~일 15:00~17:00   | 5.0    | 도시락          | 경문       |
| 아콘스틀            | 8      | 02-364-1300    | 서울 특별시 서대문구 신현역로 17 1층 110호 | 월~금 일 11:00~21:00              | NNULL             | 4.0    | 식탁           | 경문       |
| 진돈부리            | 9      | 010-4726-7684  | 서울 서대문구 신현로 18 신촌 거리 B104   | 월~토 17:00~20:20                | NNULL             | 4.0    | 시카고샤브        | 경문       |
| 이로운 밥상          | 10     | 02-6402-4910   | 서울 서대문구 연세대길 20 2층          | 월~토 일 10:30~20:00              | 월~토 일 15:00~17:00 | 3.0    | 한식           | 경문       |
| 슬로우캘리           | 11     | 02-364-1307    | 서울 서대문구 이화여대길 19 1층         | 월~금 11:00~20:00                | 월~금 15:00~17:00   | 4.0    | 아시안          | 경문       |
| 까이식당            | 12     | 070-7779-8890  | 서울 서대문구 이화여대길 24 2층         | 월~일 11:00~20:00                | NNULL             | 4.0    | 양식           | 경문       |
| 아민 이화           | 13     | 02-363-0113    | 서울 서대문구 이화여대길 52-31 1층      | 월~일 11:00~21:00                | NNULL             | 4.0    | 일식           | 경문       |
| 유소바             | 14     | 070-8224-7956  | 서울 서대문구 이화여대길 4 1층          | 월~일 11:30~21:30                | NNULL             | 4.0    | 아시안          | 경문       |
| 포히엔베트남쌀국수 이대점   | 15     | 02-365-1960    | 서울 서대문구 연세대길 29 1층          | 월~일 9:00~18:00                 | NNULL             | 4.0    | 아시안          | 경경한민족    |
| 마더락             | 16     | 02-364-1307    | 서울 특별시 서대문구 이화여대길 29 101호   | 월~금 10:30~20:00                | NNULL             | 4.0    | 한식           | 경문       |
| 티이라 파스타         | 17     | 0507-1413-4268 | 서울 서대문구 이화여대길 24 2층         | 월~금 10:30~20:00                | NNULL             | 4.0    | 양식           | 경문       |
| 스탠바이키친          | 18     | 02-365-6353    | 서울 서대문구 연대중문길 49 1층 101호    | 월~금 10:30~20:00                | NNULL             | 4.0    | 양식           | 경문       |
| 이로운 밥상          | 19     | 02-365-1245    | 서울 서대문구 연대중문길 27-3 2층       | 월~토 11:00~21:00                | 월~토 15:00~17:00   | 3.0    | 한식           | 후문       |
| 제이션 카페리아 이대후문점  | 20     | 02-364-1300    | 서울 서대문구 신현역로 18 신촌 거리 B104  | 월~일 11:00~21:00                | 월~일 15:00~17:00   | 3.0    | 한식           | 후문       |
| 헐리우드            | 21     | 02-723-4868    | 서울 서대문구 성신로 551 1층          | 월~일 10:00~21:00                | NNULL             | 3.0    | 양식           | 후문       |
| 식탁 하늬솔점         | 22     | 02-362-0777    | 서울 서대문구 성신로 527 하늬솔빌딩       | 월~일 11:00~20:00                | NNULL             | 1.0    | 양식           | 후문       |
| 불밥              | 23     | 02-362-9833    | 서울 서대문구 이화여대길 11 2층         | 월~토 11:00~22:00                | 월~토 15:00~16:30   | 3.0    | 한식           | 경문       |
| 제이션 이대점         | 24     | 02-362-9333    | 서울 서대문구 이화여대길 1 1층          | 월~금 10:00~22:00                | NNULL             | 3.0    | 한식           | 경경한민족    |
| 의원상             | 25     | 02-362-9333    | 서울 서대문구 이화여대길 72-5 1층       | 월~금 10:00~22:00                | NNULL             | 3.0    | 한식           | 경문       |
| 화리라미마리탕         | 26     | 02-313-0158    | 서울 서대문구 이화여대길 76 1층         | 월~일 10:00~22:00                | NNULL             | 3.0    | 중식           | 경문       |
| 마더린베이글          | 27     | 070-7758-3030  | 서울 서대문구 이화여대길 5 1층 1층       | 월~금 8:30~16:30, 토 10:00~16:30  | NNULL             | 4.0    | 베이커리         | 경문       |
| 청년밥상문간 이화여자대학교점 | 28     | 02-393-5460    | 서울 서대문구 이화여대길 56-8 치하 1층    | 화~일 10:30~20:00                | NNULL             | 2.0    | 중식           | 경경한민족    |
| 등촌샤브칼국수 신촌점     | 29     | 02-364-5801    | 서울 서대문구 신촌역로 43 1층          | 월~일 11:00~22:00                | 월~일 15:00~17:00   | 4.0    | 한식           | 경문       |
| 슬로우캘리 이대점       | 30     | 0507-1398-7850 | 서울 서대문구 신촌역로 43 1층          | 월~일 11:00~22:00                | 월~일 15:00~17:00   | 5.0    | 샤브샤브         | 경문       |
|                 | 31     | 0507-1391-7188 | 서울 서대문구 이화여대길 78 가동 1층      | 월~일 11:00~21:00                | 월~일 15:00~17:00   | NNULL  | 샐러드          | 경문       |

### 식당 삭제 후)

| res_name        | res_id | phone_num      | address                    | operating_hours                | break_time      | rating | cuisine_type | location |
|-----------------|--------|----------------|----------------------------|--------------------------------|-----------------|--------|--------------|----------|
| 호미곶식당           | 1      | 070-4154-2000  | 서울 서대문구 이화여대길 24 2층        | 월~금 11:00~20:30, 토 11:30~20:30 | 월~일 15:00~17:00 | 4.0    | 일식           | 경포       |
| 남아시아티           | 2      | 02-312-1238    | 서울 서대문구 이화여대길 6 1층         | 월~일 15:00~17:00                | NULL            | 4.0    | 일식           | 경포       |
| 원조오리온           | 3      | 02-313-3198    | 서울 특별시 서대문구 이화여대길 28 1층    | 월~일 10:00~21:00                | NULL            | 4.0    | 베이커리         | 경포       |
| 심플리스트           | 4      | 810-2583-3199  | 서울 서대문구 이화여대길 24 2층        | 월~일 11:30~21:30                | 월~일 15:00~17:00 | 4.0    | 양식           | 경포       |
| 초식主义            | 5      | 02-365-5679    | 서울 서대문구 이화여대길 19 1층        | 월~일 11:00~20:30                | NULL            | 4.0    | 샐러드          | 경포       |
| 아동카드교           | 6      | 02-440-1991    | 서울 서대문구 이화여대길 10 1층        | 월~일 11:00~21:00                | NULL            | 3.0    | 카페           | 경포       |
| 한국민족학교          | 7      | NULL           | 서울특별시 서대문구 신촌역길 7 1층       | 월~일 11:00~21:00                | 월~일 15:00~17:00 | 3.0    | 일식           | 경포       |
| 아몬스쿨            | 8      | 02-364-1301    | 서울특별시 서대문구 신촌역길 17 1층 110호 | 월~일 11:00~21:00                | NULL            | 5.0    | 도시락          | 경포       |
| 어반서비스           | 9      | 02-640-2049    | 서울 서대문구 이화여대길 2 201호       | NULL                           | NULL            | 4.0    | 사무실          | 경포       |
| 소아방심            | 10     | 02-640-2049    | 서울 서대문구 이화여대길 2 201호       | 월~일 10:30~20:00                | 월~일 15:00~17:00 | 3.0    | 일식           | 경포       |
| 카페카페            | 11     | 02-639-7917    | 서울 서대문구 이화여대길 56-18 1층     | 월~일 11:00~21:00                | 월~일 15:00~17:00 | 4.0    | 카페           | 경포       |
| 아민 이하           | 12     | 02-639-7919    | 서울 서대문구 이화여대길 56-18 1층     | 월~일 11:00~21:00                | 월~일 15:00~17:00 | 4.0    | 양식           | 경포       |
| 우소바             | 13     | 02-363-8113    | 서울 서대문구 이화여대길 52-31 1층     | 월~일 11:30~21:30                | 월~일 15:00~17:00 | 2.0    | 일식           | 경포       |
| 포히언베트남쌈국수 이대점   | 14     | 070-8224-7959  | 서울 서대문구 이화여대길 4 1층         | 월~일 9:00~20:00                 | 월~일 15:00~17:00 | 4.0    | 아시안          | 경포       |
| 마리아 카페          | 15     | 02-365-1985    | 서울 서대문구 이화여대길 2 1층         | 월~일 11:00~21:30                | 월~일 15:00~17:00 | 2.0    | 도시락          | 경포       |
| 마리아 카페          | 16     | 02-365-1987    | 서울 서대문구 이화여대길 2 1층         | 월~일 9:00~20:00                 | 월~일 15:00~17:00 | 2.0    | 도시락          | 경포       |
| 스텐비아키친          | 17     | 0207-1413-4268 | 서울 서대문구 연대동길 72 2층         | 월~일 11:00~21:30                | 월~일 15:00~17:00 | 3.0    | 한식           | 후포       |
| 이로운 밥상          | 18     | 02-365-6533    | 서울 서대문구 연대동길 49 1층 101호    | 월~일 10:30~20:00                | 월~일 15:00~16:00 | 4.0    | 양식           | 경포       |
| 화가와요리사 이대후문점    | 19     | 02-365-1245    | 서울 서대문구 연대동길 27-6 2층       | 월~일 11:00~21:30                | 월~일 15:00~17:00 | 3.0    | 한식           | 후포       |
| 화가와요리사 이대후문점    | 20     | 02-364-1978    | 서울 서대문구 성신여대길 339          | 월~일 11:00~20:00, 토 11:00~14:00 | NULL            | 3.0    | 일식           | 경포       |
| 화가와요리사 이대후문점    | 21     | 02-364-1980    | 서울 서대문구 성신여대길 339          | 월~일 11:00~20:00                | NULL            | 3.0    | 일식           | 경포       |
| 식탁 하니솔점         | 22     | 02-362-8777    | 서울 서대문구 성신여대길 527 하니솔빌딩    | 월~일 11:00~20:00                | NULL            | 1.0    | 양식           | 후포       |
| 불밥              | 23     | 02-362-9833    | 서울 서대문구 이화여대길 11 2층        | 월~일 11:00~22:00                | 월~일 15:00~16:30 | 3.0    | 한식           | 경포       |
| 수란 이대점          | 24     | 02-392-9333    | 서울 서대문구 이화여대길 28 1층        | 월~일 11:00~21:30                | NULL            | 3.0    | 한식           | 경포       |
| 의왕식당            | 25     | 02-362-9333    | 서울 서대문구 이화여대길 72-3 3층      | 월~일 11:00~22:00                | 월~일 15:00~16:30 | 3.0    | 중식           | 경포       |
| 마리아마리탕          | 26     | 02-313-0158    | 서울 서대문구 이화여대길 5 1층         | 월~일 8:30~16:30, 토 10:00~16:30  | NULL            | 3.0    | 한식           | 경포       |
| 마더란다베이글         | 27     | 070-7758-3838  | 서울 서대문구 이화여대길 5 지상 5층      | 월~일 11:00~20:00                | NULL            | 4.0    | 베이커리         | 경포       |
| 이화성             | 28     | 02-393-8511    | 서울 서대문구 이화여대길 58-2 지하 1층   | 월~일 10:30~20:00                | 월~일 15:00~16:00 | 2.0    | 중식           | 경포       |
| 청년발상문화 이화여자대학교점 | 29     | 0507-1344-6031 | 서울 서대문구 이화여대길 52-39 지하 1층  | 월~일 11:00~20:00                | 월~일 15:00~17:00 | 5.0    | 일식           | 사무실      |
| 증명서제작 이화여대점     | 30     | 0507-1344-7838 | 서울 서대문구 신촌역길 78 가동 1층      | 월~일 11:00~22:00                | 월~일 15:00~17:00 | NULL   | 사무실          | 경포       |
| 증명서제작 이화여대점     | 31     | 0507-1391-7188 | 서울 서대문구 이화여대길 78 가동 1층     | 월~일 11:00~21:00                | 월~일 15:00~17:00 | NULL   | 사무실          | 경포       |

## ▼ 4. 리뷰 삭제

```
===== Restaurant ===== ====== Menu =====
1. Add Restaurant | 5. Add Menu
2. Update Restaurant | 6. Update Menu
3. Search Restaurant | 7. Search Menu
4. Delete Restaurant | 8. Delete Menu

===== Review ===== ====== User =====
9. Add Review | 13. Add User
10. Update Review | 14. Update User
11. Search Review | 15. Search User
12. Delete Review | 16. Delete User

=====
17. Exit

Choose an option:
12
Enter review ID to delete: 51
Review deleted successfully.
```

- 관리자가 리뷰 삭제를 할 경우에는 해당 리뷰의 review\_id를 입력하면 해당 리뷰가 삭제된다.

### 리뷰 삭제 전)

| SELECT * FROM DB2024.Review WHERE user_id='cannes' ; |           |         |        |                                                                   |
|------------------------------------------------------|-----------|---------|--------|-------------------------------------------------------------------|
|                                                      | review_id | user_id | rating | review_content                                                    |
| 2                                                    | cannes7   | 4       | 4      | 소고기 가지덮밥 맛있긴한데 양간 비싸요ㅠㅠ                                           |
| 6                                                    | cannes7   | 4       | 4      | 가장대가 좀 있긴해도 양이 많아서 먹고나면 배불려요ㅎㅎ                                    |
| 14                                                   | cannes7   | 5       | 5      | 생면이 신선하고 족발하고 뿌드려워요ㅠㅠ 추천!!                                        |
| 17                                                   | cannes7   | 4       | 4      | 기분 좋을 때마다 드는는데 맛있어요!!                                             |
| 21                                                   | cannes7   | 4       | 4      | 기분 좋을 때마다 드는는데 맛있어요!!                                             |
| 23                                                   | cannes7   | 5       | 5      | 기분 좋을 때마다 드는는데 맛있어요!!                                             |
| 28                                                   | cannes7   | 3       | 3      | 가장 기본 맛집이에요!!                                                     |
| 38                                                   | cannes7   | 5       | 5      | 여기 너무 맛있어요!!                                                      |
| 39                                                   | cannes7   | 5       | 5      | 위에 노른자 풀 터드려서 먹으면 맛있어요!                                           |
| 40                                                   | cannes7   | 4       | 4      | 상상나니 맛있네요.                                                        |
| 42                                                   | cannes7   | 4       | 4      | 국물이 깊거든요!!                                                        |
| 47                                                   | cannes7   | 5       | 5      | 너무 맛있어요!!!! 딸꾹질한 대명사.                                             |
| 49                                                   | cannes7   | 4       | 4      | 0점에 브론지로 먹으니까 맛있고 적당히 베르루요.                                       |
| 51                                                   | cannes7   | 4       | 4      | msg 하나 없이 자연친화적인 맛이에요. 속세를 벗어나고 싶으신 분들께 추천드려요.                    |
| 54                                                   | cannes7   | 4       | 4      | 김에서 해먹은 복음밥 맛이에요. 나쁘진 않아요.                                        |
| 55                                                   | cannes7   | 4       | 4      | 평소에 미국국 되게 좋아하는데 들어 미국국이래서 시켜보았어요. 나쁘진 않지만 역시 그냥 미국국이 더 나은 것 같아요. |

### 리뷰 삭제 후)

```

mysql> SELECT * FROM DB2024_Review WHERE user_id='cannes7';
+-----+-----+-----+
| review_id | user_id | rating | review_content |
+-----+-----+-----+
2	cannes7	4	소고기 가지덮밥 맛있긴한데 약간 비싸요ㅠㅠ
6	cannes7	4	가격대가 좀 있긴해도 양이 많아서 먹고나면 배불러요ㅎㅎ
14	cannes7	5	생선이 신선하고 촉촉하고 부드러워요ㅠㅠ 추천!!
17	cannes7	5	등심보다 부드러워서 먹기 좋아요.
20	cannes7	4	기초네 처음 들어보는데 맛있고 사장님이 친절하세요.
23	cannes7	5	가격이 분식 맹길때 순대떡볶을 먹는데 여전히 맛있네요.
28	cannes7	3	연어가 너무 구워져서 안 촉촉했어요ㅠㅠ
38	cannes7	5	위에 노른자 툭 터뜨려서 먹으면 맛있어요!
40	cannes7	4	상상하니 맛있겠네요.
42	cannes7	4	저도 맛있었어요!! 맛있어요.
47	cannes7	5	나중에 맛있어요!! 단짠단짠의 대명사.
49	cannes7	4	아침에 브런치로 먹으니까 맛있고 깨닫히 배부르네요.
54	cannes7	4	절어서 해먹는 볶음밥 맛이에요. 나쁘지 않아요.
60	cannes7	4	평소에 미국국 도개 좋아하는데 들깨 미국국이라서 시켜보았어요. 나쁘진 않지만 역시 그냥 미국국이 더 나은 것 같아요.
+-----+-----+-----+
14 rows in set (0.00 sec)

```

▼ (17) 데이터베이스에 검색(select)을 하기 위한 인터페이스와 쿼리를 가지고 있어야 한다.

▼ 1. 회원정보 검색

- 내 정보 확인

```

===== My Page =====

1. My information
2. Update My information
3. Search for Other Users
4. Delete My Account
5. Exit

=====
Choose an option: 1

== My Information ==

ID: s2eojeong
Name: 조서정
Student ID: 2276305
Email: s2eojeong@gmail.com
Location: 후문

=====
```

- 다른 유저 정보 확인

```

===== My Page =====

1. My information
2. Update My information
3. Search for Other Users
4. Delete My Account
5. Exit

=====
Choose an option: 3
Enter the user ID: cannes7

== User Information ==

ID: cannes7
Name: 고은서

=====

```

## ▼ 2. 메뉴 검색

- 선택한 식당 내에 존재하는 메뉴 검색

| Name                                                                   | ID     | Phone Number   | Address                    | Operating Hours                | Break Time         | Rating | Cuisine Type | Location |
|------------------------------------------------------------------------|--------|----------------|----------------------------|--------------------------------|--------------------|--------|--------------|----------|
| <hr/>                                                                  |        |                |                            |                                |                    |        |              |          |
| 모미자식당                                                                  | 1      | 070-4154-2000  | 서울 서대문구 이화여대길 24 2층        | 월~금 11:00~20:30, 토 11:30~20:30 | 월~토 15:00~17:00    | 4.8    | 일식           | 정문       |
| 원조오우                                                                   | 3      | 02-313-3198    | 서울특별시 서대문구 이화여대길 29 1층     | 월~토 10:00~21:00                | 정보 없음              | 3.0    | 베이커리         | 정문       |
| 신플라스트                                                                  | 4      | 010-2583-3198  | 서울 서대문구 이화여대길 24 2층        | 월~토 11:30~21:30                | 월~토 15:00~17:00    | 4.0    | 양식           | 정문       |
| 초식곳간                                                                   | 5      | 02-365-5679    | 서울 서대문구 이화여대2가길 19 1층      | 월~금 11:00~19:30                | 정보 없음              | 4.0    | 샐러드          | 정문       |
| 유아케도코                                                                  | 6      | 02-6401-7991   | 서울 서대문구 이화여대5길 28 101호     | 월~토 11:00~21:00                | 정보 없음              | 3.0    | 일식           | 정문       |
| 우미마루                                                                   | 7      | 정보 없음          | 서울특별시 서대문구 신촌역 18 1층       | 월~일 11:00~21:00                | 월~일 15:00~17:00    | 3.0    | 일식           | 정문       |
| 이온스풀                                                                   | 8      | 02-364-1301    | 서울특별시 서대문구 신촌역로 17 1층 110호 | 월~금, 일 11:00~21:00             | 정보 없음              | 5.0    | 도시락          | 정문       |
| 여바웃사브                                                                  | 10     | 02-6402-9494   | 서울 서대문구 이화여대2가길 201호       | 정보 없음                          | 정보 없음              | 4.0    | 사브사브         | 정문       |
| 소오방집                                                                   | 11     | 02-6397-8917   | 서울 서대문구 이화여대길 50-10 1층     | 월~토, 일 10:30~20:00             | 월~토, 일 15:00~17:00 | 3.0    | 일식           | 정문       |
| 끼어나서당                                                                  | 12     | 070-7779-8809  | 서울 서대문구 이화여대2가길 24 1층      | 월~금 11:00~20:00                | 월~금 15:00~17:00    | 4.0    | 아시안          | 정문       |
| 이민 이화                                                                  | 13     | 02-365-0113    | 서울 서대문구 이화여대길 52-51 1층     | 월~일 11:00~21:00                | 정보 없음              | 4.0    | 양식           | 정문       |
| 포히언비트남발국수 이대점                                                          | 15     | 02-365-1985    | 서울 서대문구 이화여대3길 2 1층        | 월~일 11:00~21:00                | 정보 없음              | 4.0    | 아시안          | 정문       |
| 타이어 피스타                                                                | 17     | 0507-1413-4268 | 서울 서대문구 이화여대7길 24 2층       | 월~금 10:30~20:00                | 정보 없음              | 4.0    | 양식           | 정문       |
| 불법                                                                     | 23     | 02-362-9833    | 서울 서대문구 이화여대길 11 2층        | 월~금 11:00~22:00                | 월~토 15:00~16:30    | 3.0    | 한식           | 정문       |
| 수라 이대점                                                                 | 24     | 02-392-9333    | 서울 서대문구 이화여대2가길 20 1층      | 월~금 11:00~21:00                | 정보 없음              | 3.0    | 한식           | 정문       |
| 의원상                                                                    | 25     | 정보 없음          | 서울 서대문구 이화여대길 72-5 1층      | 월~화, 목, 금, 토, 일 10:30~21:30    | 정보 없음              | 3.0    | 중식           | 정문       |
| 페리마마라탕                                                                 | 26     | 02-313-0158    | 서울 서대문구 이화여대길 76 1층        | 월~일 10:00~22:00                | 정보 없음              | 3.0    | 중식           | 정문       |
| 청년방상장간 이화여자대학교점                                                        | 29     | 0507-1344-6831 | 서울 서대문구 이화여대길 52-39 지하 1층  | 월~금 11:00~20:00                | 월~금 16:00~17:00    | 4.0    | 한식           | 정문       |
| 트존사브락국수 신촌점                                                            | 30     | 0507-1398-7850 | 서울 서대문구 신촌역로 43 1층         | 월~일 11:00~22:00                | 월~일 15:00~17:00    | 5.0    | 사브사브         | 정문       |
| <hr/>                                                                  |        |                |                            |                                |                    |        |              |          |
| ==== Search Detail ====                                                |        |                |                            |                                |                    |        |              |          |
| 1. Search Restaurant Menu   2. Search Restaurant Reviews   3. Exit     |        |                |                            |                                |                    |        |              |          |
| <hr/>                                                                  |        |                |                            |                                |                    |        |              |          |
| Choose an option: 1   Enter Restaurant ID: 8   Menu ID Menu Name Price |        |                |                            |                                |                    |        |              |          |
| <hr/>                                                                  |        |                |                            |                                |                    |        |              |          |
| 1                                                                      | 아콘스풀김밥 | 3900           |                            |                                |                    |        |              |          |
| 2                                                                      | 손대떡볶음  | 3900           |                            |                                |                    |        |              |          |

- 사용자가 식당 검색 후, 해당 식당 내에 존재하는 메뉴들만을 조회해볼 수 있다.

## ▼ 3. 식당 검색

- cuisine\_type별로 식당 검색

| =====                                                              |               |                        |                                                |                    |        |          |  |
|--------------------------------------------------------------------|---------------|------------------------|------------------------------------------------|--------------------|--------|----------|--|
| 1. Search By Category<br>2. Search By Other Information<br>3. Exit |               |                        |                                                |                    |        |          |  |
| =====                                                              |               |                        |                                                |                    |        |          |  |
| Choose an option: 1<br>Enter Cuisine Type: 일식                      |               |                        |                                                |                    |        |          |  |
| Restaurant Name                                                    | Phone Number  | Address                | Operating Hours                                | Break Time         | Rating | Location |  |
| 모미지식당                                                              | 070-4154-2800 | 서울 서대문구 이화여대7길 24 2층   | 월~금 11:00~20:30, 토 11:30~20:30 월~토 15:00~17:00 | 4.0                |        | 정문       |  |
| 낭만식탁                                                               | 02-312-1238   | 서울 서대문구 이화여대5길 6 1층    | 월~토 11:00~20:00                                | 월~토 15:00~17:00    | 정보 없음  | 정문       |  |
| 유아케도쿄                                                              | 02-6401-7991  | 서울 서대문구 이화여대3길 28 101호 | 월~토 11:00~21:00                                | 정보 없음              | 3.0    | 정문       |  |
| 우미마루                                                               | 정보 없음         | 서울특별시 서대문구 신촌역로 18 1층  | 월~일 11:00~21:00                                | 월~일 15:00~17:00    | 3.0    | 정문       |  |
| 소오밥집                                                               | 02-6397-8917  | 서울 서대문구 이화여대길 50-10 1층 | 월~토, 일 10:30~20:00                             | 월~토, 일 15:00~17:00 | 3.0    | 정문       |  |
| 우소바                                                                | 070-8224-7956 | 서울 서대문구 이화여대2길 4 1층    | 월~일 11:30~21:30                                | 월~일 15:00~17:00    | 2.0    | 정문       |  |
| 회가와요리사 이대후문점                                                       | 02-364-1970   | 서울 서대문구 성산로 539        | 월~금 11:00~20:00, 토 11:00~14:00                 | 정보 없음              | 3.0    | 후문       |  |

- cuisine\_type별로 검색하기에서 '일식', '한식', '중식' 등 type 명을 입력하면 해당 cuisine\_type에 해당하는 식당들을 조회할 수 있다.

- 복합적인 조건으로 식당 검색

| =====                                            |    |                |                           |                 |                 |        |              |
|--------------------------------------------------|----|----------------|---------------------------|-----------------|-----------------|--------|--------------|
| Choose an option: 2                              |    |                |                           |                 |                 |        |              |
| Enter Restaurant Name (or press Enter to skip):  |    |                |                           |                 |                 |        |              |
| Enter Cuisine Type (or press Enter to skip): 한식  |    |                |                           |                 |                 |        |              |
| Enter Location (or press Enter to skip): 정문      |    |                |                           |                 |                 |        |              |
| Enter Minimum Rating (or press Enter to skip): 3 |    |                |                           |                 |                 |        |              |
| Restaurants found:                               |    |                |                           |                 |                 |        |              |
| Name                                             | ID | Phone Number   | Address                   | Operating Hours | Break Time      | Rating | Cuisine Type |
| 불법                                               | 23 | 02-562-7833    | 서울 서대문구 이화여대8길 11 2층      | 월~토 11:00~22:00 | 월~토 15:00~16:30 | 3.0    | 한식           |
| 수라 이데일                                           | 24 | 02-592-9333    | 서울 서대문구 이화여대2길 20 1층      | 월~금 11:00~21:00 | 정보 없음           | 3.0    | 한식           |
| 청년발상단간 이화여자대학교점                                  | 29 | 0507-1344-6031 | 서울 서대문구 이화여대길 52-59 지하 1층 | 월~금 11:00~28:00 | 월~금 16:00~17:00 | 4.0    | 한식           |

- 다른 복합적인 조건 (식당 이름, 위치, 최소 평점) 으로 원하는 식당을 조회할 수 있다.

- 랜덤으로 식당 검색

| =====                                     |    |               |                     |                 |                 |        |              |
|-------------------------------------------|----|---------------|---------------------|-----------------|-----------------|--------|--------------|
| Choose an option: 2                       |    |               |                     |                 |                 |        |              |
| Randomly recommended restaurants for you! |    |               |                     |                 |                 |        |              |
| -----                                     |    |               |                     |                 |                 |        |              |
| Name                                      | ID | Phone         | Address             | Operating hour  | Break time      | Rating | Cuisine Type |
| 우소바                                       | 14 | 070-8224-7956 | 서울 서대문구 이화여대2길 4 1층 | 월~일 11:30~21:30 | 월~일 15:00~17:00 | 2.0    | 일식           |
|                                           |    |               |                     |                 |                 |        | 정문           |

- 어디를 갈지 고민이 될 땐 랜덤으로 식당 1개를 추천받을 수 있다.

## ▼ 4. 리뷰 검색

| =====                                                                |    |              |                       |                 |            |        |              |
|----------------------------------------------------------------------|----|--------------|-----------------------|-----------------|------------|--------|--------------|
| 1. Search by Restaurant Name, Cuisine Type, Location, Minimum Rating |    |              |                       |                 |            |        |              |
| 2. Search by Cuisine Type                                            |    |              |                       |                 |            |        |              |
| 3. Exit                                                              |    |              |                       |                 |            |        |              |
| =====                                                                |    |              |                       |                 |            |        |              |
| Choose an option: 1                                                  |    |              |                       |                 |            |        |              |
| Enter Restaurant Name (or press Enter to skip): 초식곳간                 |    |              |                       |                 |            |        |              |
| Enter Cuisine Type (or press Enter to skip):                         |    |              |                       |                 |            |        |              |
| Enter Location (or press Enter to skip):                             |    |              |                       |                 |            |        |              |
| Enter Minimum Rating (or press Enter to skip):                       |    |              |                       |                 |            |        |              |
| Restaurants found:                                                   |    |              |                       |                 |            |        |              |
| Name                                                                 | ID | Phone Number | Address               | Operating Hours | Break Time | Rating | Cuisine Type |
| 초식곳간                                                                 | 5  | 02-365-5679  | 서울 서대문구 이화여대2가길 19 1층 | 월~금 11:00~19:30 | 정보 없음      | 4.0    | 샐러드          |
|                                                                      |    |              |                       |                 |            |        | 정문           |

```

===== Search Detail =====

1. Search Restaurant Menu
2. Search Restaurant Reviews
3. Exit

=====
Choose an option:
2
Enter restaurant ID: 5
Review ID User ID      User Name      Rating      Review Content
-----
15     chacha091      차현주        4          정말 불곰이 반할만하네요.
14     cannes7        고은서        5          생연에 신선하고 촉촉하고 부드러워요ㅠㅠ 추천!!
13     meanwest       김민서        5          리코타치즈 들어간 샌드위치 꼭 드세요. 신세계입니다

```

- 사용자가 식당을 검색한 후, Search Detail에서 원하는 식당의 res\_id를 입력하면 해당 식당에 대한 리뷰들을 조회할 수 있다.

## ▼ 7. 요구조건 외의 팀만의 강점

### ▼ 직관적이고 사용하기 쉬운 UI 디자인

- E-MATEASY는 사용자 인터페이스(UI)를 텍스트와 콘솔 기반으로 구현하였지만, 사용자의 편의를 극대화하기 위해 직관적이고 사용하기 쉽게 구성하였다. 단순한 텍스트 기반의 UI에서도 사용자가 원하는 기능을 쉽게 찾고 이용할 수 있도록 설계하여, 사용자 경험을 최적화했다.
  - 회원가입 시, 올바른 email 형식으로 작성할 수 있도록 예시 형식 구현
  - 메뉴 검색 시, 식당 목록을 보고 검색할 수 있도록 구현
  - 리뷰 등록 시, 리뷰 id는 자동으로 입력되도록 구현
  - 리뷰 등록 시, 메뉴 목록을 보고 해당하는 메뉴에 리뷰를 등록할 수 있도록 구현
  - 회원탈퇴 시 본인의 password를 입력하고 일치하는 시에만 삭제하도록 구현

### ▼ 사용자 입력을 위한 친화적인 인터페이스

#### ▼ 복합적인 조건을 통한 검색 기능

```

=====
1. Search by Restaurant Name, Cuisine Type, Location, Minimum Rating
2. Search by Cuisine Type
3. Exit

=====
Choose an option: 1
Enter Restaurant Name (or press Enter to skip): 
Enter Cuisine Type (or press Enter to skip): 양식
Enter Location (or press Enter to skip): 청문
Enter Minimum Rating (or press Enter to skip): 3
Restaurants found:
Name           ID    Phone Number      Address          Operating Hours      Break Time      Rating      Cuisine Type      Location
심플리스트      4     010-2583-3190    서울 서대문구 이화여대길 24 2층      월~토 11:30~21:30      월~토 15:00~17:00      4.0       양식      정문
어린 이화      13     02-363-0113    서울 서대문구 이화여대길 52-31 1층      월~일 11:00~21:00      정보 없음      4.0       양식      정문
티아라 파스타    17     0507-1413-4268    서울 서대문구 이화여대길 24 2층      월~금 10:30~20:00      정보 없음      4.0       양식      정문

```

- 사용자가 복합적인 조건을 통해 검색이 가능하도록 시스템을 설계하였다.
- 예를 들어, 식당을 검색할 때 식당 이름, 위치, 최소 평점 등을 사용자가 원하는 조건만 여려 조합으로 입력하면, 그에 맞는 식당들을 출력해준다.
- 이러한 다중 조건 검색 기능을 통해 사용자는 자신이 원하는 조건을 충족하는 식당들을 선택적으로 찾아볼 수 있다.

### ▼ 선택적 수정 기능

| <유저가 회원정보를 수정할 때>

```

===== My Page =====

1. My information
2. Update My information
3. Search for Other Users
4. Delete My Account
5. Exit

=====
Choose an option: 2

===== Update My information =====

New Password (Press enter to skip): danchoo
New Name (Press enter to skip):
New Student ID (Press enter to skip):
New Email (Press enter to skip): danchoo@gmail.com
New Location (Press enter to skip): 정문

Update successful!

```

<관리자가 식당 정보를 수정할때>

```

31          슬로우캘리 이대점
-----
Enter Restaurant ID: 31
Enter New Restaurant Name (Press enter to skip): 패스트캘리
Enter New Phone Number (Press enter to skip):
Enter New Address (Press enter to skip): 서울 서대문구 이화여대길 52 나동 2층
Enter New Operating Hours (Press enter to skip):
Enter New Break Time (Press enter to skip): 월~일 15:00~16:30
Enter New Rating (Press enter to skip):
Enter New Cuisine Type (Press enter to skip):
Enter New Location (Press enter to skip):
Restaurant updated successfully.

```

- 사용자는 데이터를 수정할 때 원하는 부분만 선택적으로 수정할 수 있도록 설계되었다.
- 예를 들어, 수정하지 않는 부분은 <Enter> 키를 눌러 넘어가면 해당 필드는 변경되지 않으며, 바꾸고 싶은 필드만 골라서 변경 할 수 있다.
- 이러한 선택적 수정 기능은 사용자에게 더욱 유연하고 편리한 데이터 관리 환경을 제공한다.

#### ▼ 입력 검증 및 데이터 무결성 유지

- 조건에 맞지 않은 사용자 입력이 주어졌을 때 데이터베이스에 잘못된 정보가 저장/ 출력되지 않도록 하는 기능을 구현하였다. 시스템은 이를 검증하고 오류 메시지를 반환하여 해당 정보가 없음을 알린다.
  - 회원가입 시 형식이 올바르지 않은 이메일 입력

```

=====
1. User Login
2. User Register
3. Admin Login
4. Exit

=====
Choose an option: 2
New userID: ewha123
New password: ewha123
Name: 김아화
Student ID: 1871086
Email (format: example@domain.com): ewha123@gmail.com
Please enter a valid email address (ex: example@domain.com).
Email (format: example@domain.com): ewha123@gmail.com
Please enter a valid email address (ex: example@domain.com).
Email (format: example@domain.com): ewha123@gmail.com
1. 정문
2. 후문
Choose an Location (Enter 1 or 2): 1
Registration successful!

```

◦ 존재하지 않는 식당 ID로 메뉴/리뷰를 검색

| Name            | ID | Phone Number   | Address                   | Operating Hours | Break Time      | Rating | Cuisine Type | Location |
|-----------------|----|----------------|---------------------------|-----------------|-----------------|--------|--------------|----------|
| 불밥              | 23 | 02-362-9833    | 서울 서대문구 이화여대길 11 2층       | 월~토 11:00~22:00 | 월~토 15:00~16:30 | 3.0    | 한식           | 정문       |
| 수박 이대점          | 24 | 02-392-9333    | 서울 서대문구 이화여대2가길 28 1층     | 월~금 11:00~21:00 | 정보 없음           | 3.0    | 한식           | 정문       |
| 청년밥상문간 이화여자대학교점 | 29 | 0507-1344-6031 | 서울 서대문구 이화여대길 52-39 지하 1층 | 월~금 11:00~20:00 | 월~금 16:00~17:00 | 4.0    | 한식           | 정문       |

```

==== Search Detail ====
1. Search Restaurant Menu
2. Search Restaurant Reviews
3. Exit

=====
Choose an option: 1
Enter Restaurant ID: 35
Not existing restaurant.

```

◦ 존재하지 않는 메뉴 ID에 리뷰를 작성

```

===== Search menu =====
Enter Restaurant ID: 14
Menu ID Menu Name      Price
-----
1        우삼겹정식        16000
2        마구로정식        16000

===== Add review =====
Enter User ID: s2eojeong
Enter Restaurant ID: 14
Enter Menu ID: 3
Enter Rating (1-5): 5
Enter review comment: 더운 여름엔 역시 소바가 최고!!
Review not added: Menu ID does not exist for the given restaurant.

```

◦ 회원가입 시, 이미 존재하는 userID를 사용

```

=====
1. User Login
2. User Register
3. Admin Login
4. Exit

=====
Choose an option: 2
New userID: s2eojeong
New password: hehe
Name: 조서정복제인간
Student ID: 2276305
Email (format: example@domain.com): s2eojeong@gmail.com
1. 정문
2. 후문
Choose an Location (Enter 1 or 2): 1
Registration failed. Duplicated userID. Please choose a different one.

```

- 회원탈퇴 시, 해당 회원이 작성한 리뷰들도 함께 삭제

- 삭제 전)

```

mysql> SELECT * FROM DB2024_Review WHERE user_id='s2eojeong';
+-----+-----+-----+-----+
| review_id | user_id | rating | review_content
+-----+-----+-----+-----+
| 1 | s2eojeong | 5 | 유회당밥 너무 고소하고 맛있어용!!
| 7 | s2eojeong | 0 | 스모크드 베이컨 할리피뇨 바게트 매콤하긴한데 맛있게 잘 먹었어요 추천!!
| 12 | s2eojeong | 3 | 홍새우 처음 먹어보는데 나쁘지 않아요.
| 26 | s2eojeong | 4 | 학교 주변에 사케동 파는 곳 많은데 여기도 나쁘지 않은 것 같아요.
| 33 | s2eojeong | 4 | 양이 약간 적긴해도 나쁘지 않네요.
| 39 | s2eojeong | 5 | 둘이 먹다 하나 죽어도 모를 맛이에요... 강추
| 41 | s2eojeong | 5 | 잠시동안 베트남에 있다왔네요.
| 48 | s2eojeong | 5 | 파스타 중에서 오일 파스타를 제일 좋아하는데 맛있네요 ㅎㅎ
| 52 | s2eojeong | 3 | 나물이 약간 질겨서 씹는데 불편했어요ㅠㅠ
+-----+-----+-----+-----+
9 rows in set (0.00 sec)

```

- 삭제 후)

```

===== My Page =====

1. My information
2. Update My information
3. Search for Other Users
4. Delete My Account
5. Exit

=====
Choose an option: 4

Are you sure you want to delete your account?
All data related to you will be deleted.

Enter 'y' if you want to delete, or 'n': y
Please enter your PASSWORD: s2eojeong
Membership withdrawal completed.

```

```
mysql> SELECT * FROM DB2024_Review WHERE user_id='s2eojeong';
Empty set (0.00 sec)
```

## ▼ 보안성 강화

```
===== My Page =====

1. My information
2. Update My information
3. Search for Other Users
4. Delete My Account
5. Exit

=====
Choose an option: 3
Enter the user ID: cannes7

== User Information ==

ID: cannes7
Email: cannes7@ewhain.net
```

- 사용자가 다른 유저의 정보를 검색할 때 다른 유저의 개인정보(user\_pw, student\_id)들은 확인할 수 없도록 했다.
- 따라서 user\_id, email만 확인할 수 있다.

## ▼ 8. 팀 구성원 담당 부분

| 이름  | SQL                                                                                                                                                                                                                                             | Java code                                                                                                                                                                                                                                                                                                                | Report                                        | 발표                                                | 기타                                                       |  |
|-----|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------|---------------------------------------------------|----------------------------------------------------------|--|
| 고은서 | - 테이블 작성(DB2024_Review)                                                                                                                                                                                                                         | -DB2024TEAM07_AdminMain<br>-DB2024TEAM07_MenuManager<br>-DB2024TEAM07_RestaurantManager<br>-DB2024TEAM07_ReviewManager 등은 제외<br>-DB2024TEAM07_ReviewManager 활용 안<br>-DB2024TEAM07_RestaurantDAO의 getAllRestaurants() 추가<br>-DB2024TEAM07_MenuDAO의 getAllMenuByRestaurant() 추가<br>-DB2024TEAM07_UserDAO의 getAllUsers() 추가 |                                               | - 레포트 4, 6번 항목 작성                                 | - 기밀 ppt 제작                                              |  |
| 김민서 | - create.sql은 구조 작성(테이블 통합 및 잡다 코드/존속)<br>- 테이블 작성(DB2024_User, DB2024_Menu, DB2024_Rating)<br>- 별 작성(DB2024_OtherUser, DB2024_UserReview, DB2024_ResReview)<br>- 인덱스 작성(DB2024_idx_AvgRating, DB2024_idx_Review)<br>- 초기 데이터 추가(DB2024_Rating) | - 프로젝트 구조 폴더<br>-DB2024TEAM07_User<br>-DB2024TEAM07_UserDAO<br>-DB2024TEAM07_ReviewVO<br>-DB2024TEAM07_RatingDAO<br>-DB2024TEAM07_ReviewDTO<br>-DB2024TEAM07_ReviewDAO<br>-DB2024TEAM07_UserReviewDTO<br>-DB2024TEAM07_RestReviewVO                                                                                      | - 레포트 4, 6번 항목 작성                             | - 중간 발표 및 대본 작성                                   |                                                          |  |
| 조서정 | - 테이블 작성(DB2024_Restaurant)<br>-인덱스 작성(DB2024_idx_Menu, DB2024_idx_Restaurant)<br>-초기 데이터 삽입(DB2024_User, DB2024_Restaurant, DB2024_Menu, DB2024_Review)                                                                                        | - 프로젝트 구조 폴더<br>-DB2024TEAM07_Menu<br>-DB2024TEAM07_MenuDAO<br>-DB2024TEAM07_Restaurant<br>-DB2024TEAM07_RestaurantDAO                                                                                                                                                                                                   | - 레포트 총 편집 (사진 갈피, 설명 등)<br>- 레포트 1~8 번 항목 작성 |                                                   | - 초기 앤디 데이터 수집<br>- 더미 데이터 추가                            |  |
| 차현주 | - Schema Diagram 제작<br>- 월레이션 스키마 제작<br>- 테이블 추가 (DB2024_Review_Menu, Res_Mapping)                                                                                                                                                              | -DB2024TEAM07_ReviewManager<br>-DB2024TEAM07_AdminMain 리뷰 메뉴 추가<br>-DB2024TEAM07_ReviewDAO 수정<br>-DB2024TEAM07_RatingDAO getAvg() 수정                                                                                                                                                                                     | - 레포트 4, 6번 항목 작성                             | - 중간 ppt 제작                                       | - 요구영세서 작성                                               |  |
| 한시강 | - ER Diagram 제작                                                                                                                                                                                                                                 | - 프로젝트 구조 폴더<br>-DB2024TEAM07_Menu<br>-DB2024TEAM07_MenuMain<br>-DB2024TEAM07_UserManager<br>-DB2024TEAM07_ReviewManager<br>-DB2024TEAM07_ReviewDAO addReview() 수정<br>-DB2024TEAM07_RestaurantDAO의 getRandomRestaurant() 추가<br>-DB2024TEAM07_RestaurantManager의 displayRandomRestaurant() 추가<br>-프로젝트 전체 javadoc 주석 작성   | - 레포트 4, 6번 항목 작성                             | - 기밀 발표 및 대본 작성<br>- 중간 ppt 수정<br>- 프로그램 시연 영상 제작 | - 요구영세서 작성<br>- 요구영세서 수정본 작성<br>- 프로젝트 팀장<br>- README 추가 |  |