HW9 – DB Application Programming

Department Name: 정보컴퓨터공학부
202055650
이윤재

Submission date: 2024-12-15

The report should include the following:

1. 생성한 테이블 설명

- HW8에 작성한 ER 다이어그램을 바탕으로, 어떤 테이블을 생성하였는지 설명함

Brand
- 호텔 체인은 여러 개의 brand를 소유한다.
- brand_id를 주키로 갖는 테이블 생성한다.

```sql
CREATE TABLE Brand (
    brand_id SERIAL PRIMARY KEY,
    name VARCHAR(15) UNIQUE NOT NULL
);
```

Hotel
- 호텔 브랜드는 여러 지점(branch)을 갖는다.
- hotel_id를 주키로 갖는 테이블을 생성한다.
- brand_id는 외래키로 Brand 테이블을 참조한다.

```sql
CREATE TABLE Hotel (
    hotel_id SERIAL PRIMARY KEY,
    name VARCHAR(20) NOT NULL,
    location VARCHAR(15),
    brand_id INT REFERENCES Brand(brand_id) ON DELETE CASCADE
);
```

Room
- 하나의 호텔 지점에는 여러 개의 객실이 존재한다.
- 호텔 지점 안에 여러 개의 객실이 존재함을 표현하기 위해 지점마다 고유하게 갖는 hotel_id를 외래키로 활용한다.
- 객실마다 고유한 객실 ID를 가지며, 1박당 가격, 객실에 대한 설명이 포함 되어있다.

```sql
CREATE TABLE Room (
    room_id SERIAL PRIMARY KEY,
    hotel_id INT REFERENCES Hotel(hotel_id) ON DELETE CASCADE,
    price NUMERIC(10, 2) NOT NULL,
    description VARCHAR(30)
);
```

Customer

- 고객은 자신의 고유한 ID(customer_id)를 갖는다.
- 이름, 이메일 전화번호가 속성에 포함된다.

```
CREATE TABLE Customer (
    customer_id SERIAL PRIMARY KEY,
    first_name VARCHAR(50) NOT NULL,
    last_name VARCHAR(50) NOT NULL,
    email VARCHAR(100),
    phone VARCHAR(20)
);
```

Reservation

- 고객의 ID, 객실 ID를 활용하여 예약 투플을 생성한다.
- 예약은 자신의 고유한 ID(reservation_id)를 갖는다.
- 체크인 날짜와 체크아웃 날짜를 기반으로 계산된 숙박 예상 비용을 기록해 둔다.

```
CREATE TABLE Reservation (
    reservation_id SERIAL PRIMARY KEY,
    customer_id INT REFERENCES Customer(customer_id) ON DELETE CASCADE,
    room_id INT REFERENCES Room(room_id) ON DELETE CASCADE,
    check_in_date DATE NOT NULL,
    check_out_date DATE NOT NULL,
    total_price NUMERIC(10, 2)
);
```

Housekeeper

- 하우스 키퍼는 자신의 고유한 ID(hk_id)를 갖는다.
- status 속성값을 활용하여 현재 하우스키퍼가 일을 하고 있는지 쉬고 있는지를 판단한다.
- 하우스 키퍼의 현재 상태를 나타내는 status 속성은 Boolean타입으로 나타낸다.

```
CREATE TABLE Housekeeper (
    hk_id SERIAL PRIMARY KEY,
    status BOOLEAN,
    name VARCHAR(20)
);
```

check_in

- 현재 고객이 투숙 중인 객실의 정보를 얻기 위한 클래스이다.
- 고객정보(customer_id), 객실정보(room_id)를 외래키로 갖는다.
- housekeeper_id를 통해 하우스키퍼가 배정되었는지를 확인할 수 있다.

```
CREATE TABLE check_in (
    check_in_id SERIAL PRIMARY KEY,
    customer_id INT,
    room_id INT,
    fee NUMERIC(10, 2),
    check_in DATE NOT NULL,
    check_out_date DATE NOT NULL,
    housekeeper_id INT,
    FOREIGN KEY (customer_id) REFERENCES Customer(customer_id),
    FOREIGN KEY (room_id) REFERENCES Room(room_id),
    FOREIGN KEY (housekeeper_id) REFERENCES housekeeper(hk_id)
);
```

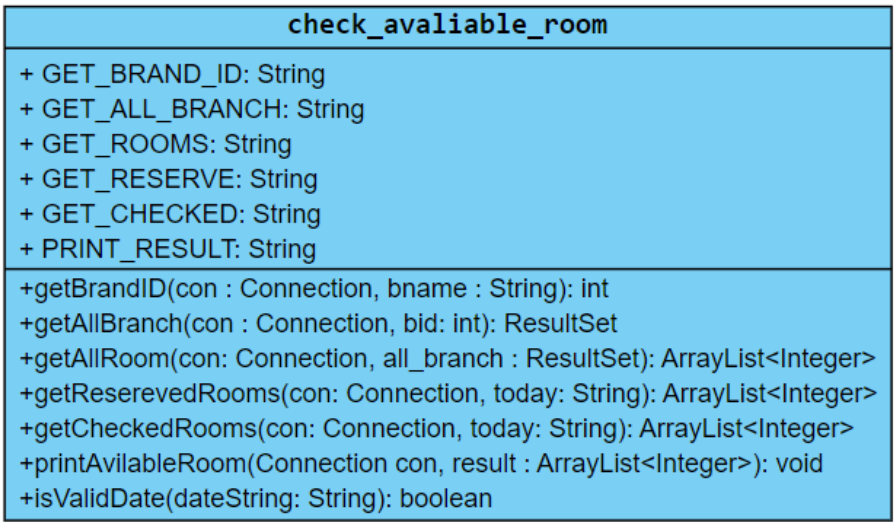
2. 구현한 클래스 설명

- C/C++이나 JAVA 언어를 사용하여 구현한 클래스에 대한 다이어그램 포함
- 핵심 함수들에 대하여 설명함

[사용자]

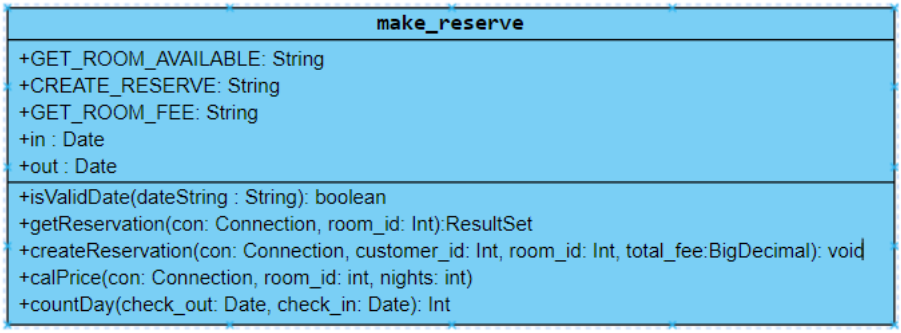
check_available_room

- 일반 사용자 모드용 질의 rooms_available을 위한 클래스
- 호텔 브랜드 이름과 체크인 날짜를 입력해 예약할 수 있는 객실을 조회한다.
- get_brandID 함수를 통해 입력받은 호텔 브랜드의 이름을 브랜드 ID(brand_id)로 변환한다.
- get_all_branch 함수를 통해 해당 브랜드의 모든 지점(branch) 정보를 가져온다.
- get_every_room 함수를 통해 모든 지점의 객실 정보를 가져온다.
- get_reserved_rooms 함수를 통해 예약 정보를 가져온다. 입력한 날짜에 체크인이 가능한지를 확인하기 위해 필요한 정보이다.
- print_available_room 함수를 통해 해당 호텔 브랜드에 속하면서 입력한 날짜에 체크인이 가능한 모든 객실 정보를 출력한다.
- isValidDate 함수를 통해 입력한 형식이 날짜 형식이 맞는지를 검사한다.



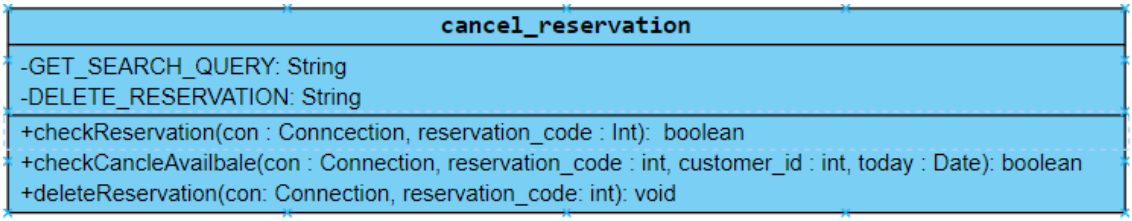
make_reserve

- 일반 사용자용 트랜잭션 reserve(make a reservation)를 위한 클래스
- 예약하려는 객실의 ID(room_id)와 체크인 날짜, 체크아웃 날짜를 입력한다.
- 날짜 형식이 유효한지, 숙박 기간이 유효한지(체크아웃 날짜가 체크인 날짜 이후인지)를 확인
- isValidDate함수를 통해 입력한 형식이 날짜 형식이 맞는지를 검사한다.
- getAvailableRoom 함수를 통해 해당 객실에 대한 예약 내역을 모두 가져온다.
- createReservation을 통해 입력한 정보를 바탕으로 예약을 생성한다.
- countDay 함수와 calPrice 함수를 통해 요금을 계산한다.
- 클래스의 메인 함수 내에서 예약이 가능한지를 파악한다. (한 방은 하루에 한 예약자에 의해서만 예약이 가능하다.)



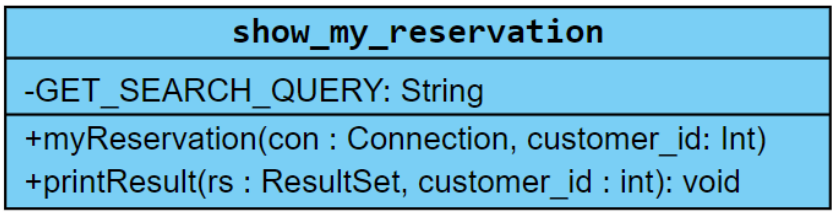
cancel_reservation

- 일반 사용자용 트랜잭션 cancel(cancel a reservation)을 수행하기 위한 클래스
- 예약 번호를 이용해 자신의 예약을 취소한다.
- 취소하는 날짜(시스템 날짜, LocalDate.now()활용)가 체크인 이전이어야 취소가 가능하다.



show_my_reservation

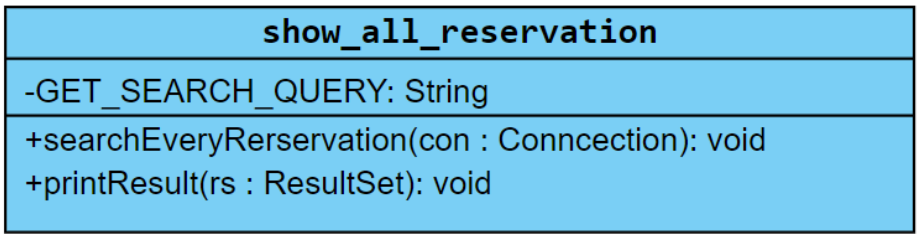
- 자신의 예약 현황을 조회한다.
- myReservation 함수를 통해 자신의 예약 정보를 가져온다.
- printResult 함수를 통해 가져온 자신의 예약 정보를 출력한다.



[추가기능]

show_all_reservation

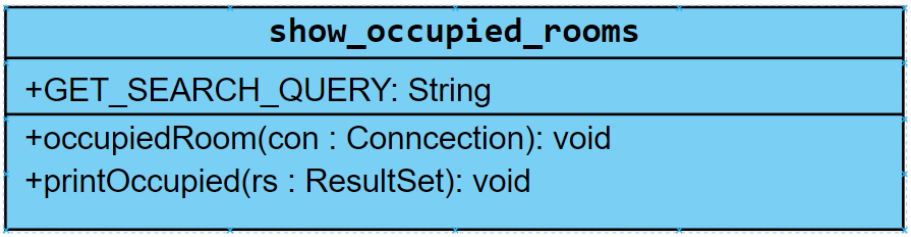
- 호텔에 존재하는 모든 객실에 대한 정보를 조회하는 클래스
- 사용자는 해당 호텔에 존재하는 모든 객실에 대한 조회가 가능하다.
- search_every_room 함수를 통해 해당 호텔에서 운영 중인 모든 객실을 가져온다.
- print_result 함수를 통해 search_every_room의 결과를 출력한다.
- get_hotel_name 함수를 통해 room_id를 이용해 호텔의 정보(이름, 장소)를 가져온다.



[관리자]

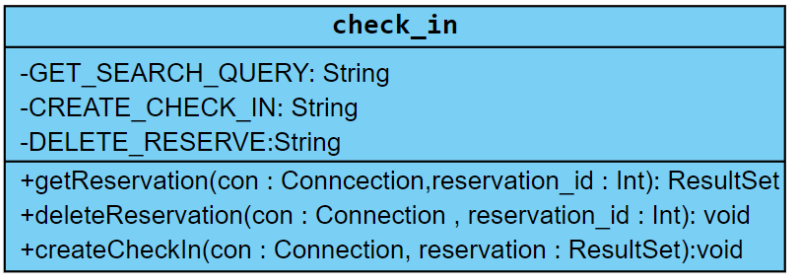
show_occupied_rooms

- 관리자 모드용 질의 rooms_occupied(view the currently occupied rooms)을 위한 클래스
- 현재 이용 중인 객실과 고객의 정보를 출력한다.
- occupiedRoom 함수를 통해 check_in 테이블에 있는 모든 정보를 가져온다.
- printOccupied 함수를 통해 가져온 정보를 출력한다.



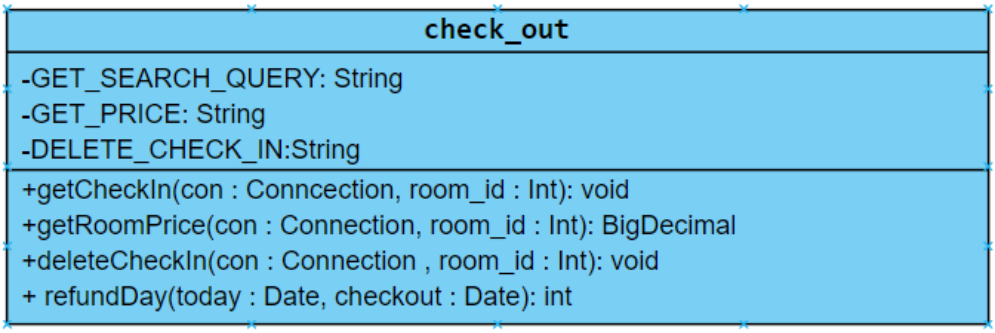
check_in

- 관리자 모드용 트랜잭션 check_in(check-in a guest)을 위한 클래스
- 예약 번호를 통해 체크인을 수행
- 현재 날짜와 체크인 날짜가 동일해야 체크인 수행이 가능하다.
- getReservation 함수를 통해 예약을 조회한다.
- deleteReservation 함수를 통해 해당 예약 내역을 삭제하고 createCheckIn 함수를 활용해 check_in 릴레이션에 새로운 데이터를 생성한다.



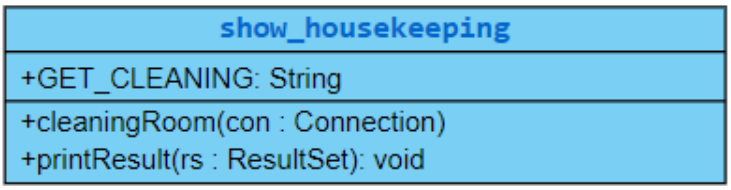
Check_out

- 관리자 모드용 트랜잭션 check_out(check-out a guest)을 위한 클래스
- 객실 ID를 통해 체크아웃을 수행한다.
- 체크아웃 시, 최종 결제 금액을 계산한다.
- 체크아웃 예정일 보다 일찍 체크아웃하는 경우 환불을 수행한다.
- refundDay 함수를 통해 체크아웃 날짜보다 얼마나 일찍 체크아웃하는지를 계산한다.



show_housekeeping

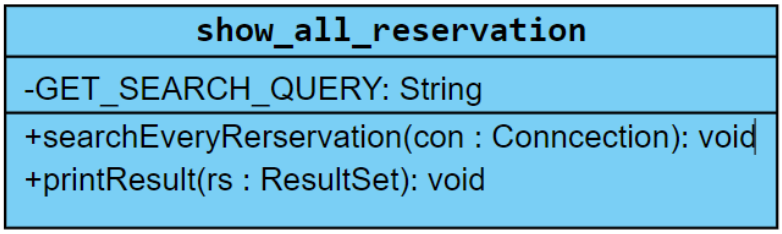
- 관리자 모드용 질의 housekeeping(list house-keeping assignments)을 위한 클래스
- 하우스키퍼에 의해 관리되고 있는 모든 객실의 정보를 출력한다.
- cleaningRoom함수를 통해 하우스키퍼가 배치되어 있는 객실을 조회한다.
- printCleaningRoom함수를 통해 해당 정보를 출력한다.



[추가기능]

show_all_reservation

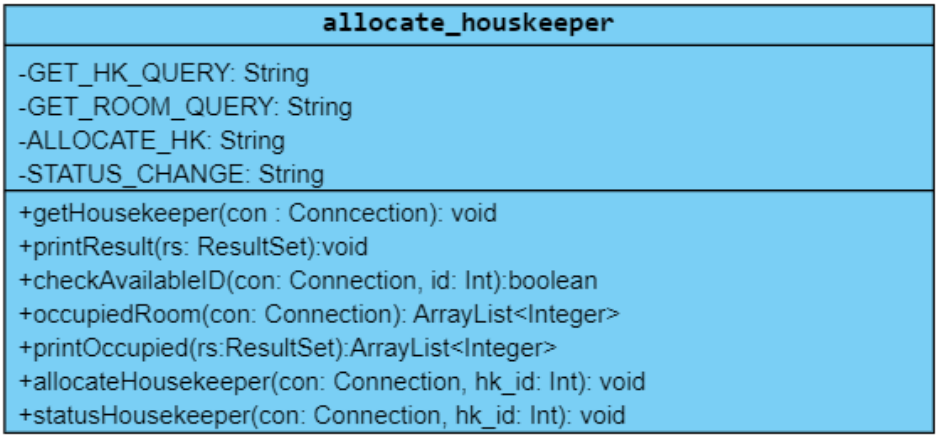
- 모든 예약정보를 출력
- searchEveryRerservation함수를 통해 모든 예약 정보를 가져온다.
- printResult함수를 통해 searchEveryRerservation를 통해 가져온 모든 예약 정보를 출력한다.



[추가기능]

Allocate_housekeeper

- 체크인 상태의 객실에 하우스 키퍼를 배정한다.
- getHosusekeeper함수를 통해 일하고 있지 않은 하우스키퍼를 조회한다.
- occupiedRoom 함수를 통해 하우스 키퍼가 배정되지 않은 방을 조회한다.
- 유효한 하우스키퍼 ID와 객실ID정보를 입력 받아 객실에 하우스키퍼를 배정한다.



3. 구현한 기능 설명

숙제에서 제시한 각 기능에 대해서, 어떤 함수에서 어떤 방식으로 구현하였는지 코드를 제시하면서 설명함

[로그인]

관리자모드: login('manager')

- 관리자 모드의 경우 해당 모든 권한 있는 'yunjaelee' ROLE로 데이터베이스에 연결
- if문을 활용해 입력된 아이디의 값이 'manager'인 경우, 관리자 모드로 로그인

```
public final static String DB_DRIVER_CLASS = "org.postgresql.Driver";
public final static String DB_URL = "jdbc:postgresql://localhost:5432/lee55650";
public final static String DB_USERNAME = "yunjaelee";
public final static String DB_PASSWORD = "love3928";

public static Connection managerLogin() throws ClassNotFoundException, SQLException {
    Connection con = null;
    // load the Driver Class
    Class.forName(DB_DRIVER_CLASS);

    // create the connection now
    con = DriverManager.getConnection(DB_URL, DB_USERNAME, DB_PASSWORD);
    return con;
}
```

사용자모드: login('customer', unique_id)

- 관리자 모드의 경우 제한되는 권한을 갖는 'customer' ROLE로 데이터베이스에 연결
- if문을 활용해 입력된 아이디의 값이 'customer'인 경우, 사용자 모드로 로그인
- 추가로, customer_id를 입력 받아 해당 id를 갖는 투플이 customer 릴리이션에 존재하는지 확인

```
public final static String CUSTOMER_DB_DRIVER_CLASS = "org.postgresql.Driver";
public final static String CUSTOMER_DB_URL = "jdbc:postgresql://localhost:5432/lee55650";
public final static String CUSTOMER_DB_USERNAME = "customer";
public final static String CUSTOMER_DB_PASSWORD = "customer";

public static Connection customerLogin() throws ClassNotFoundException, SQLException {

    Connection con = null;
    // load the Driver Class
    Class.forName(CUSTOMER_DB_DRIVER_CLASS);

    // create the connection now
    con = DriverManager.getConnection(CUSTOMER_DB_URL, CUSTOMER_DB_USERNAME, CUSTOMER_DB_PASSWORD);
    return con;
}
```

```
main [Java Application] C:₩Program Files₩Java₩jdk-2
========== 호텔에 오신걸 환영합니다. ==========
ID : customer
Customer ID: 5
========== 사용자 메뉴 ==========
1 : OO 호텔의 모든방을 조회하기
2 : 예약가능한 객실 유형 및 비용 확인하기
3 : 예약하기
4 : 예약 취소하기
5 : 예약 조회하기
0 : 프로그램 종료
메뉴 선택:
```

```
Console ×
<terminated> main [Java Application] C:₩Program Fi
========== 호텔에 오신걸 환영합니다. ==========
ID : customer
Customer ID: 12
ID가 12인 고객이 존재하지 않습니다.
```

[사용자]

rooms_available

- view the room types and costs that are still available
- 호텔 브랜드와 체크인 날짜를 입력해 해당 날짜에 체크인이 가능한 해당 호텔 브랜드의 모든 객실을 조회한다.

1. 입력한 호텔 브랜드의 모든 지점의 모든 객실을 가져온다.

```
public static ArrayList<Integer> getAllRoom(Connection con, ResultSet all_branch) throws SQLException {
    ArrayList<Integer> rooms = new ArrayList<Integer>();
    while (all_branch.next()) {
        PreparedStatement stmt = con.prepareStatement(GET_ROOMS);
        stmt.setInt(1, all_branch.getInt(1));
        ResultSet rs = stmt.executeQuery();
        while (rs.next()) {
            rooms.add(rs.getInt(1));
        }
    }
    return rooms;
}
```

2. 체크인 날짜에 예약 중인 객실 또는 체크인 되어있는 객실의 정보를 가져온다.

```
public static ArrayList<Integer> get_resereved_rooms(Connection con, String today) throws SQLException {
    ArrayList<Integer> rooms = new ArrayList<Integer>();
    PreparedStatement stmt = con.prepareStatement(GET_RESERVE);
    stmt.setDate(1, Date.valueOf(today));
    stmt.setDate(2, Date.valueOf(today));
    ResultSet rs = stmt.executeQuery();
    while (rs.next()) {
        rooms.add(rs.getInt("room_id"));
    }
    return rooms;
}

public static ArrayList<Integer> get_checked_rooms(Connection con, String today) throws SQLException {
    ArrayList<Integer> rooms = new ArrayList<Integer>();
    PreparedStatement stmt = con.prepareStatement(GET_CHECKED);
    stmt.setDate(1, Date.valueOf(today));
    stmt.setDate(2, Date.valueOf(today));
    ResultSet rs = stmt.executeQuery();
    while (rs.next()) {
        rooms.add(rs.getInt("room_id"));
    }
    return rooms;
}
```

3. 차집합 연산을 통해 입력한 체크인 날짜에 체크인이 가능한 객실을 계산한다.

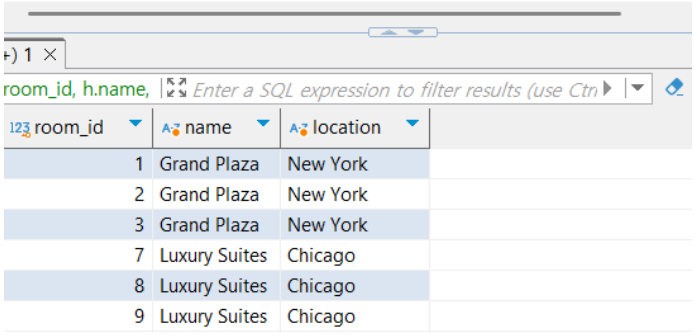
```
//예약된방
ArrayList<Integer> result4 = get_resereved_rooms(con, today);
//체크인된방
ArrayList<Integer> result5 = get_checked_rooms(con, today);

// 5. 3과 4,5에 대해서 차집합수행
result3.removeAll(result4);
result3.removeAll(result5);
```

[실행결과]

```
예약이 가능한 모든 방을 조회합니다.
호텔 브랜드 이름을 입력해주세요 : Luxury Stay
체크인 날짜를 입력해주세요 (연도-월-일): 2024-12-15
room_id    hotel_name                location                price          description
-------------------------------------------------------------------------------------------------------
1          Grand Plaza               New York                250.00         Ocean view suite
2          Grand Plaza               New York                200.00         City view room
3          Grand Plaza               New York                350.00         Luxury suite with Jacuzzi
8          Luxury Suites             Chicago                 250.00         Executive suite
```

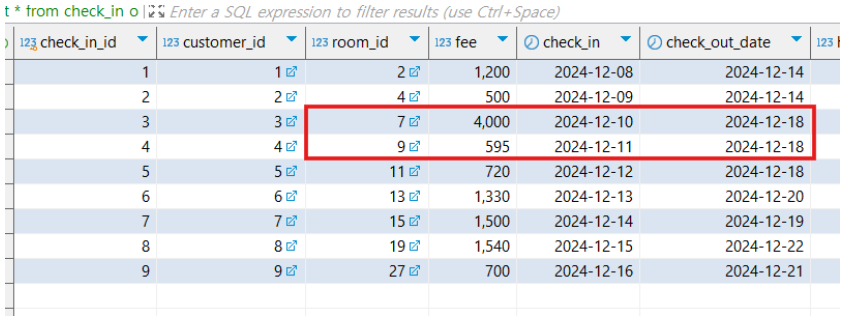
Luxury Stay에서 운영중인 모든 방

```
select room_id, h.name, h.location
from brand b, hotel h , room r
where b.brand_id = h.brand_id and H.hotel_id = R.hotel_id
and b.name = 'Luxury Stay';
```



room_id	name	location
1	Grand Plaza	New York
2	Grand Plaza	New York
3	Grand Plaza	New York
7	Luxury Suites	Chicago
8	Luxury Suites	Chicago
9	Luxury Suites	Chicago

7, 9번 방의 경우, 입력한 날짜(2024-12-15)에 해당 객실을 이용중인 고객이 있어 조회가 되지 않았음을 알 수 있다.



check_in_id	customer_id	room_id	fee	check_in	check_out_date
1	1	2	1,200	2024-12-08	2024-12-14
2	2	4	500	2024-12-09	2024-12-14
3	3	7	4,000	2024-12-10	2024-12-18
4	4	9	595	2024-12-11	2024-12-18
5	5	11	720	2024-12-12	2024-12-18
6	6	13	1,330	2024-12-13	2024-12-20
7	7	15	1,500	2024-12-14	2024-12-19
8	8	19	1,540	2024-12-15	2024-12-22
9	9	27	700	2024-12-16	2024-12-21

cost_at_checkout

- calculate the total cost for the guest at checkout time
- 예약을 생성할 때, 숙박 기간 동안의 숙박비를 계산한다.
- 체크아웃이 빠른 경우에는 체크아웃이 빠른 일 수만큼 환불을 해준다.
- 체크인 테이블에서 투플을 삭제하면서 체크아웃이 완료된다.

1. 예약을 생성할 때, 숙박 기간 동안의 숙박비를 계산해 reservation 테이블에 저장을 해둔다.

- countDay함수를 통해 숙박기간을 검색하고, claPrice함수를 통해 해당 기간동안의 숙박비용을 계산한다. 예상 숙박 금액은 Reservation 테이블의 한 속성으로 저장된다.

```
System.out.println("예약을 진행합니다");
int nights = countDay(in, out);

BigDecimal fee = calPrice(con, room_id, nights);
System.out.println("예상 숙박 금액은 " + fee + "입니다.");
// 2. 조건을 충족하면 예약을 생성하기
createReservation(con, customer_id, room_id, fee);
```

2. 예약된 날짜보다 체크아웃이 빠른 경우, 해당 일 수만큼 환불을 수행해준다. 체크아웃 수행이 종료된 이후에 지불해야 하는 금액을 출력한다.

- check_out 클래스에서 refundDay함수를 통해 체크아웃이 며칠 빨리 일어났는지를 계산한다.
- 조기 체크아웃시, 해당 일 수만큼의 금액을 제외하고 숙박비를 계산한 후, 그 결과를 콘솔에 출력한다.

```
int day = refundDay(today_date, check_out);
total_price = price + (getRoomPrice(con, room_id).intValue() * day);
System.out.println("지불하실 금액은 " + total_price + "원 입니다.");
```

3. 체크인 테이블에서 해당 투플이 삭제하면서 체크아웃이 완료된다.

```
// 2-2.check_in에서 데이터를 삭제하기
deleteCheckIn(con, room_id);
con.commit();
```

```
public static void deleteCheckIn(Connection con, int room_id) throws SQLException {
    PreparedStatement stmt = con.prepareStatement(DELETE_CHECK_IN);
    stmt.setInt(1, room_id);
    stmt.executeUpdate();
    System.out.print("체크아웃이 완료되었습니다.");
}
```

my_reservations

- list future reservations for the guest
- 나의 예약 정보를 출력한다.

1. customer_id를 통해 자신의 예약현황을 모두 조회하고, printResult함수를 통해 조회한 예약 현황을 모두 출력한다.

```
public static void myReservation(Connection con, int customer_id) throws SQLException {
    PreparedStatement stmt = con.prepareStatement(GET_SEARCH_QUERY);
    stmt.setInt(1, customer_id);
    ResultSet rs = stmt.executeQuery();
    printResult(rs, customer_id);
    stmt.close();
}
```

[실행결과]

```
<terminated> show_my_reservation [Java Application] C:₩Program Files₩Java₩jdk-21₩bin₩javaw.exe  (2024. 12. 15. 오전
고객 번호가 10인 고객님의 예약 내역을 확인합니다.
reservation_id       room_id     check_in             check_out            price
-------------------------------------------------------------------------------------
10                   10          2024-12-24           2024-12-30           1000.00
-------------------------------------------------------------------------------------
```

reserve

- make a reservation
- 객실ID, 체크인 날짜, 체크 아웃 날짜를 입력해 해당 기간에 객실예약이 가능한지를 조회한다.

1. 입력한 체크인 날짜, 체크 아웃 날짜에 해당 객실에 예약이 가능한지를 확인한다. 이를 위해, 예약현황과 체크인 현황을 모두 확인한다.

```
// 1. 해당 room_id에 대한 예약을 조회하고, 체크인 날짜, 체크아웃 날짜 확인하기
ResultSet rs = getReservation(con, room_id);
System.out.println("Valid input");
boolean flag = true;

in = Date.valueOf(check_in);
out = Date.valueOf(check_out);

while(rs.next()) {
    if((out.compareTo(rs.getDate("check_in_date")) <= 0) || (in.compareTo(rs.getDate("check_out_date")) >= 0)) {}
    else {
        System.out.println("해당 기간에 대한 예약이 이미 존재합니다.");
        flag = false;
        break;
    }
};
```

2. 해당 기간에 객실 사용이 가능하다면 예약을 생성한다. 예약을 생성할 때 숙박기간을 활용하여 미리 숙박요금을 미리 계산해 두고, 해당 내용을 테이블에 저장한다.

```
// 요금 계산하기
public static BigDecimal calPrice(Connection con, int room_id, int nights) throws SQLException {
    PreparedStatement stmt = con.prepareStatement(GET_ROOM_FEE);
    stmt.setInt(1, room_id);
    ResultSet rs = stmt.executeQuery();
    rs.next();
    int price = rs.getInt(1);
    BigDecimal fee = BigDecimal.valueOf(price).multiply(BigDecimal.valueOf(nights));
    return fee;
}

public static void createReservation(Connection con, int customer_id, int room_id, BigDecimal total_fee)
        throws SQLException {

    PreparedStatement stmt = con.prepareStatement(CREATE_RESERVE);

    stmt.setInt(1, customer_id);
    stmt.setInt(2, room_id);
    stmt.setDate(3, in);
    stmt.setDate(4, out);
    stmt.setBigDecimal(5, total_fee);
    stmt.executeUpdate();

    System.out.println(in + "-" + out + "기간에 " + room_id + " 번 방이 예약되었습니다.");
    stmt.close();
}
```

[실행결과]

```
<terminated> make_reserve [Java Application] C:₩Program Files₩Java₩jdk-21₩bin₩javaw.exe
1 : 예약가능한 객실 유형 및 비용 확인하기
2 : 예약진행하기
메뉴 선택: 2
예약을 위해 객실ID, 체크인 날짜, 체크아웃 날짜를 입력해주세요
ROOM_ID : 3
Check in Date : 2024-12-24
Check out Date : 2024-12-27
Valid input
예약을 진행합니다
숙박 기간은 3일 입니다.
예상 숙박 금액은 1050입니다.
2024-12-24-2024-12-27기간에 3 번 방이 예약되었습니다.
```

```
select * from reservation r order by r.customer_id;
```

vation 1 ×

t * from reservation | Enter a SQL expression to filter results (use Ctrl+Space)

reservation_id	customer_id	room_id	check_in_date	check_out_date	total_price
1	1	11	2024-12-25	2024-12-29	1,200
2	2	12	2024-12-26	2024-12-30	380
16	3	3	2024-12-24	2024-12-27	1,050
3	3	13	2024-12-27	2024-12-29	900
15	3	11	2024-12-14	2024-12-18	1,200
5	5	15	2024-12-29	2024-12-31	1,100
6	6	6	2024-12-20	2024-12-25	1,700

[추가기능] 예약하기전에 비어 있는 객실을 조회할 수 있도록 구현

```
<terminated> make_reserve [Java Application] C:₩Program Files₩Java₩jdk-21₩bin₩javaw.exe  (2024. 12. 15. 오전 10:59:32 – 오전 10:59:51) [pid: 4000]
1 : 예약가능한 객실 유형 및 비용 확인하기
2 : 예약진행하기
메뉴 선택: 1
예약이 가능한 모든 방을 조회합니다.
호텔 브랜드 이름을 입력해주세요 : Luxury Stay
체크인 날짜를 입력해주세요 (연도-월-일): 2024-12-24
room_id    hotel_name                location                price          description
---------------------------------------------------------------------------------------------------------
1          Grand Plaza               New York                250.00         Ocean view suite
2          Grand Plaza               New York                200.00         City view room
3          Grand Plaza               New York                350.00         Luxury suite with Jacuzzi
```

cancel

- cancel a reservation
- 예약 번호를 통해 예약 취소를 수행한다.
- 당일예약은 취소가 불가능한다.

1. 예약 ID와 사용자 ID를 통해 예약 내역을 확인한다.

```
public static boolean checkReservation(Connection con, int reservation_code, int customer_id) throws SQLException {
    boolean result = true;
    PreparedStatement stmt = con.prepareStatement(GET_RESERVATION);
    stmt.setInt(1, reservation_code);
    stmt.setInt(2, customer_id);
    ResultSet rs = stmt.executeQuery();
    if (!rs.next()) {
        result = false;
    }
    stmt.close();
    return result;
}
```

2. 예약 취소 요건을 만족시키는 조사 후, 예약 취소 동작을 수행한다.

- checkCancleAvailbale함수를 통해 예약 취소 가능 여부를 판단한다.
- 예약이 존재하고, 취소 조건을 만족한다면 deleteReservation 함수를 통해 reservation 테이블에 존재하는 예약 내역을 삭제한다.

```
if (rs.next() && (rs.getDate("check_in_date").compareTo(today) > 0)) {
    result = true;
}
stmt.close();
return result;
```

```java
// 1. 예약 번호로 예약을 찾고
if (checkReservation(con, reservation_code, customer_id)) {
    // 2. check_in 이전이면 delete
    if (checkCancleAvailbale(con, reservation_code, customer_id, Date.valueOf(today))) {
        // System.out.println("해당 예약 삭제 동작");
        deleteReservation(con, reservation_code);
    } else {
        System.out.println("취소가 불가능한 예약입니다.");
        System.out.println("체크인 날짜 이전에 취소가 가능합니다.");
    }
} else {
    System.out.println("예약 내역이 존재하지 않습니다.");
}
con.commit();
```

[결과]

```
Console ×
<terminated> cancel_reservation [Java Application] C:₩Program Files₩Java₩jdk-21₩
취소하려는 예약의 예약번호를 입력하세요 :16
예약(예약번호: 16)이 취소되었습니다.
```

예약번호 16에 해당하는 투플이 삭제되었음을 알 수 있다.

	reservation_id	customer_id	room_id	check_in_date	check_out_date	total_price
1	15	3	11	2024-12-14	2024-12-18	1,200
2	10	10	10	2024-12-24	2024-12-30	1,000
3	9	9	9	2024-12-23	2024-12-26	260
4	8	8	8	2024-12-22	2024-12-26	800
5	7	7	7	2024-12-21	2024-12-27	650
6	6	6	6	2024-12-20	2024-12-25	1,700
7	5	5	15	2024-12-29	2024-12-31	1,100

체크인 날짜가 지난 예약은 취소가 불가능하다. 오늘 날짜(2024-12-15)이전의 예약은 취소가 불가능하다.

```
Console ×
<terminated> cancel_reservation [Java Application] C:₩Program Files₩Java₩jdk-21₩bin₩javav
취소하려는 예약의 예약번호를 입력하세요 :15
취소가 불가능한 예약입니다.
체크인 날짜 이전에 취소가 가능합니다.
```

```
select * from reservation r order by r.reservation_id d  Enter a SQL expression to filter results (use Ctrl+Space)
```

	reservation_id	customer_id	room_id	check_in_date	check_out_date	total_price
1	15	3	11	2024-12-14	2024-12-18	1,200
2	10	10	10	2024-12-24	2024-12-30	1,000
3	9	9	9	2024-12-23	2024-12-26	260
4	8	8	8	2024-12-22	2024-12-26	800
5	7	7	7	2024-12-21	2024-12-27	650

[추가기능] 해당 호텔 브랜드에 포함된 모든 객실을 조회 가능하도록 함

[결과]

현재 우리 호텔에서 운영중인 객실 정보입니다.

room_id	hotel_name	location	price	description
1	Grand Plaza	New York	250.00	Ocean view suite
2	Grand Plaza	New York	200.00	City view room
3	Grand Plaza	New York	350.00	Luxury suite with Jacuzzi
4	Budget Inn	Los Angeles	90.00	Single room with desk
5	Budget Inn	Los Angeles	110.00	Double room with queen bed
6	Budget Inn	Los Angeles	130.00	Family room
7	Luxury Suites	Chicago	220.00	Deluxe suite
8	Luxury Suites	Chicago	250.00	Executive suite
9	Luxury Suites	Chicago	180.00	Standard room
10	Comfort Inn	San Francisco	100.00	Economy single room
11	Comfort Inn	San Francisco	120.00	Standard double room
12	Comfort Inn	San Francisco	140.00	Triple room
13	Elite Resort	Miami	300.00	Oceanfront suite
14	Elite Resort	Miami	250.00	Presidential suite
15	Elite Resort	Miami	190.00	Superior room
16	Budget Lodge	Seattle	70.00	Basic room
17	Budget Lodge	Seattle	90.00	Standard room
18	Budget Lodge	Seattle	120.00	Suite with balcony
19	Premium Villa	Hawaii	500.00	Beachfront villa
20	Premium Villa	Hawaii	550.00	Luxury villa with private pool
21	Premium Villa	Hawaii	400.00	Poolside suite
22	Urban Retreat	Dallas	150.00	Business suite
23	Urban Retreat	Dallas	180.00	Deluxe room with city view
24	Urban Retreat	Dallas	220.00	Luxury suite
25	Business Inn	Boston	85.00	Economy single room
26	Business Inn	Boston	110.00	Family room
27	Business Inn	Boston	140.00	Triple room
28	City Center Hotel	San Diego	200.00	Luxury suite with ocean view
29	City Center Hotel	San Diego	230.00	Penthouse suite
30	City Center Hotel	San Diego	180.00	Superior double room

[관리자]

rooms_occupied

- view the currently occupied rooms

-체크인 테이블을 활용해 현재 고객들이 이용중인 객실의 정보를 얻는다.

1. 현재 체크인 상태 즉, 고객들이 이용중인 객실의 정보를 가져오고, printOccupied함수를 통해 해당 정보를 출력한다.

```
public static final String GET_SEARCH_QUERY = "SELECT * FROM check_in";

public static void occupiedRoom(Connection con) throws SQLException {
    PreparedStatement stmt = con.prepareStatement(GET_SEARCH_QUERY);
    ResultSet rs = stmt.executeQuery();
    printOccupied(rs);
    stmt.close();
}
```

[결과]

```
현재 고객님이 이용중인 모든 방을 조회합니다.
customer_id          room_id    check_in_date        check_out_date
------------------------------------------------------------------------
6                    13         2024-12-13           2024-12-20
7                    15         2024-12-14           2024-12-19
8                    19         2024-12-15           2024-12-22
3                    7          2024-12-10           2024-12-18
4                    9          2024-12-11           2024-12-18
5                    11         2024-12-12           2024-12-18
9                    27         2024-12-16           2024-12-21
------------------------------------------------------------------------
```

Housekeeping

- list house-keeping assignments
- check_in 테이블에 housekeeper_id 속성을 이용하여 해당 객실에 하우스키퍼가 배정되어있는지를 확인한다. Housekeeper_id는 null값을 가질 수 있으며, 외래키로서 housekeeper테이블을 참조한다.
- check_in된 방에 하우스 키퍼를 할당한다.

1. check_in 테이블에서 housekeeper_id가 널 값이 아닌 경우를 모두 가져와 출력한다.

```
public static final String GET_CLEANING = "SELECT * FROM check_in where housekeeper_id is NOT NULL";

public static void cleaningRoom(Connection con) throws SQLException {
    PreparedStatement stmt = con.prepareStatement(GET_CLEANING);
    ResultSet rs = stmt.executeQuery();
    printCleaningRoom(rs);
    stmt.close();
}
```

[결과]

```
하우스 키퍼가 관리하고 있는 방을 조회합니다.
room_id    check_in_date        check_out_date       housekeeper
------------------------------------------------------------------
7          2024-12-10           2024-12-18           3
9          2024-12-11           2024-12-18           7
11         2024-12-12           2024-12-18           5
27         2024-12-16           2024-12-21           10
------------------------------------------------------------------
```

check_in

- check-in a guest
- 예약을 기반으로 체크인을 수행한다.
- reservation_id를 활용해 체크인을 수행하고, 현재 날짜와 check_in 날짜가 동일해야 체크인을 수행할 수 있다.

1. reservation_id를 입력해 해당 ID에 해당하는 예약이 존재하는지를 확인하고 예약 정보를 가져온다.

```
public static ResultSet getReservation(Connection con, int reservation_id) throws SQLException {
    PreparedStatement stmt = con.prepareStatement(GET_SEARCH_QUERY);
    stmt.setInt(1, reservation_id);
    ResultSet rs = stmt.executeQuery();
    rs.next();
    return rs;
}
```

2. 입력한 reservation_id에 대한 예약이 존재하고, 해당 예약의 체크인 날짜와 현재 날짜가 동일하다면 체크인을 수행한다. 체크인 table에 새로운 투플을 삽입하면서 체크인 동작을 수행한다.

```
if (reservation.getDate("check_in_date").equals(Date.valueOf(today))) {
    System.out.println("체크인이 가능합니다.");
    createCheckIn(con, reservation);
```

```
public static void createCheckIn(Connection con, ResultSet reservation) throws SQLException {
    PreparedStatement stmt = con.prepareStatement(CREATE_CHECK_IN);
    stmt.setInt(1, reservation.getInt("customer_id"));
    stmt.setInt(2, reservation.getInt("room_id"));
    stmt.setInt(3, reservation.getInt("total_price"));
    stmt.setDate(4, reservation.getDate("check_in_date"));
    stmt.setDate(5, reservation.getDate("check_out_date"));
    stmt.executeUpdate();
    stmt.close();
}
```

3. 체크인 수행이 완료된 예약 정보는 reservation테이블에서 삭제한다.

```
// reservation에 있는 데이터 삭제하기
public static void deleteReservation(Connection con, int reservation_id) throws SQLException {
    PreparedStatement stmt = con.prepareStatement(DELETE_RESERVE);
    stmt.setInt(1, reservation_id);
    stmt.executeUpdate();
    stmt.close();
}
```

[결과]

체크인 테스트를 위하여 오늘(2024-12-15)에 체크인을 하는 예약을 생성

	reservation_id	customer_id	room_id	check_in_date	check_out_date	total_price
1	17	3	21	2024-12-15	2024-12-19	1,600

예약번호를 기준으로 체크인 수행

```
Console ×
<terminated> check_in [Java Application] C:₩Program Files₩Java₩jdk-21₩bin₩jav
예약 번호를 입력하세요 : 17
체크인이 가능합니다.
체크인이 완료되었습니다.
```

예약정보를 바탕으로 체크인이 수행된다.

check_in 1 ×

select * from check_in o | Enter a SQL expression to filter results (use Ctrl+Space)

	check_in_id	customer_id	room_id	fee	check_in	check_out_date	housekeeper_id
1	3	3	7	4,000	2024-12-10	2024-12-18	3
2	4	4	9	595	2024-12-11	2024-12-18	7
3	5	5	11	720	2024-12-12	2024-12-18	5
4	6	6	13	1,330	2024-12-13	2024-12-20	[NULL]
5	7	7	15	1,500	2024-12-14	2024-12-19	[NULL]
6	8	8	19	1,540	2024-12-15	2024-12-22	[NULL]
7	14	3	21	1,600	2024-12-15	2024-12-19	[NULL]
8	9	9	27	700	2024-12-16	2024-12-21	10

예약테이블에서는 해당 예약번호에 해당하는 내역이 삭제된다.

reservation 1 ×

select * from reservation r order by r.reservation_id d | Enter a SQL expression to filter results (use Ctrl+Space)

	reservation_id	customer_id	room_id	check_in_date	check_out_date	total_price
1	15	3	11	2024-12-14	2024-12-18	1,200
2	10	10	10	2024-12-24	2024-12-30	1,000
3	9	9	9	2024-12-23	2024-12-26	260
4	8	8	8	2024-12-22	2024-12-26	800
5	7	7	7	2024-12-21	2024-12-27	650

체크인 날짜가 오늘(2024-12-15)이 아닌 경우에는 체크인을 할 수 없다.

ct * from reservation r order by r.reservation_id d | Enter a SQL expression to filter results (use Ctrl+Space)

	reservation_id	customer_id	room_id	check_in_date	check_out_date	total_price
	5	5	15	2024-12-29	2024-12-31	1,100
	3	3	13	2024-12-27	2024-12-29	900
	2	2	12	2024-12-26	2024-12-30	380

Console ×

<terminated> check_in [Java Application] C:₩Program Files₩Java₩jdk-21₩bin₩javaw.exe

예약 번호를 입력하세요 : 5

체크인이 불가능합니다.

check_out

- 하나의 객실은 하루에 한 손님에게만 제공됨으로 객실 ID를 활용해 체크아웃을 수행할 수 있다.
- 체크아웃 시 최종 숙박 비용을 계산한다.
- 예약할 때 체크아웃 날짜 보다 이른 경우 해당 날짜만큼 금액을 환불해준다.

1. 해당 room_id로 체크아웃이 가능한지 조회한다.]

```
// 1. room_id로 check_out이 가능한지 조회하고
ResultSet check_in_info = getCheckIn(con, room_id);

// transaction
con.setAutoCommit(false);
```

2. 예약된 날짜보다 체크아웃이 빠른 경우, 해당 일 수만큼 환불을 수행해준다.

- check_out 클래스에서 refundDay함수를 통해 체크아웃이 며칠 빨리 일어났는지를 계산한다.
- 환불 금액을 포함하여 최종 숙박 비용을 계산한다

```
Date today_date = Date.valueOf(today);
Date check_out = check_in_info.getDate("check_out_date");

int day = refundDay(today_date, check_out);
total_price = price + (getRoomPrice(con, room_id).intValue() * day);
System.out.println("지불하실 금액은 " + total_price + "원 입니다.");
```

3. 체크인 테이블에서 해당 room_id를 갖는 투플을 삭제한다.

```
public static void deleteCheckIn(Connection con, int room_id) throws SQLException {
    PreparedStatement stmt = con.prepareStatement(DELETE_CHECK_IN);
    stmt.setInt(1, room_id);
    stmt.executeUpdate();
    System.out.print("체크아웃이 완료되었습니다.");
}
```

객실 ID를 통해 체크아웃을 할 수 있으며, 체크아웃 시간이 빠르면 그에 해당하는 기간 만큼 숙박비가 제외됩니다.

```
<terminated> check_out [Java Application] C:₩Program Files₩Java₩jdk-2
객실 ID :11
지불하실 금액은 360원 입니다.
체크아웃이 완료되었습니다.
```

4	4	9	595	2024-12-11	2024-12-18
5	5	11	720	2024-12-12	2024-12-18
6	6	13	1,330	2024-12-13	2024-12-20
7	7	15	1,500	2024-12-14	2024-12-19
8	8	19	1,540	2024-12-15	2024-12-22

체크아웃이 성공적으로 동작하면 check in 테이블에서 해당 데이터가 삭제된다.

check_in_id	customer_id	room_id	fee	check_in	check_out_date
4	4	9	595	2024-12-11	2024-12-18
6	6	13	1,330	2024-12-13	2024-12-20
7	7	15	1,500	2024-12-14	2024-12-19
8	8	19	1,540	2024-12-15	2024-12-22
14	3	21	1,600	2024-12-15	2024-12-19

mark_serviced

- left join을 활용하여 모든 객실의 서비스 상태를 조회한다.

```
public static final String GET_SERVICED = "select r.room_id, ci.customer_id, ci.check_out_date "
        + "from room r left join check_in ci "
        + "ON r.room_id = ci.room_id "
        + "order by r.room_id" ;
```

[결과]

```
모든 객실의 상태를 출력합니다.
room_id    customer_id          check_out_date
-------------------------------------------------
1          null                 null
2          null                 null
3          null                 null
4          null                 null
5          null                 null
6          null                 null
7          null                 null
8          null                 null
9          4                    2024-12-18
10         null                 null
11         null                 null
12         null                 null
13         6                    2024-12-20
14         null                 null
15         7                    2024-12-19
16         null                 null
```

[추가 기능]

Housekeeper allocation

- 관리자는 하우스 키퍼를 객실에 할당할 수 있다.
- 현재 업무가 없는 하우스 키퍼를 조회한 후, 하우스 키퍼가 배정되지 않은 객실에 하우스 키퍼를 배정한다.
- 하우스 키퍼의 ID와 객실 ID를 이용해 하우스 키퍼를 할당하는데, 입력된 ID가 모두 유효한지를 확인한다.

1. 현재 업무가 없는 하우스 키퍼를 조회한 후, 하우스 키퍼가 배정되지 않은 객실 정보를 가져온다.

```
// 1. 쉬고 있는 하우스 키퍼 정보 가져오기
con = DBConnection.customerLogin();
getHousekeeper(con);

// 2.check_in되어 있는 객실 중, 하우스 키퍼가 배정되지 않은 방의 정보를 가져온다.
ArrayList<Integer> occupiedRooms = occupiedRoom(con);
```

2. 객실 ID와 하우스 키퍼 ID를 입력 받고, 입력 받은 값이 유효한 값인지를 확인한다.

- 입력한 하우스 키퍼ID를 갖는 하우스 키퍼가 어떠한 방에도 할당되어 있지 않은지를 확인하고, 입력한 객실 ID가 현재 체크인 되어 있는 ID인지를 확인한다.

```java
if (checkAvailableID(con, hkid)) {
    int roomid = sc.nextInt();
    boolean checkroom = false;
    for (int i = 0; i < occupiedRooms.size(); i++) {
        if (occupiedRooms.get(i) == roomid) {
            checkroom = true;
            break;
        }
    }
    if (checkroom) {
        allocateHousekeeper(con, hkid, roomid);

    } else {
        System.out.println("해당 RoomID를 가진 객실에는 하우스키퍼 배정이 불가합니다.");
    }
} else {
    System.out.println("해당 ID를 가진 하우스키퍼는 배정이 불가합니다.");
}
//입력한 하우스 키퍼ID에 대한 유효성 검사
public static boolean checkAvailableID(Connection con, int id) throws SQLException {
    boolean result = false;
    Statement stmt = con.createStatement();
    ResultSet rs = stmt.executeQuery(GET_HK_QUERY);
    while (rs.next()) {
        int target = rs.getInt("hk_id");
        if (target == id) {
            result = true;
            break;
        }
    }
    return result;
}
```

[결과]

```
현재 쉬고 있는 하우스키퍼 목록을 조회합니다.
hkId        name
------------------------------
2           Nancy Harris
4           Tom Lewis
6           Alice Scott
8           Linda King
1           Mary Jones
9           Paul Adams
------------------------------
청소가 되어있지 않은 모든 방을 조회합니다.
room_id     check_in_date         check_out_date        housekeeper
-------------------------------------------------------------
15          2024-12-14            2024-12-19            null
19          2024-12-15            2024-12-22            null
21          2024-12-15            2024-12-19            null
13          2024-12-13            2024-12-20            null
-------------------------------------------------------------
하우스 커퍼의 ID를 입력하세요 : 9
하우스키퍼를 배정할 객실의 ID를 입력하세요 :13
객실(객실ID: 13)에 하우스키퍼(사번: 9)가 배정되었습니다.
```

하우스 키퍼가 관리하고 있는 방을 조회합니다.

room_id	check_in_date	check_out_date	housekeep
9	2024-12-11	2024-12-18	7
27	2024-12-15	2024-12-21	10
13	2024-12-13	2024-12-20	9

[추가기능]

- 관리자는 모든 예약을 조회할 수 있음

모든 예약을 조회합니다.

reservation_id	room_id	customer_id	check_in	check_out	price
1	11	1	2024-12-25	2024-12-29	1200.00
2	12	2	2024-12-26	2024-12-30	380.00
3	13	3	2024-12-27	2024-12-29	900.00
5	15	5	2024-12-29	2024-12-31	1100.00
6	6	6	2024-12-20	2024-12-25	1700.00
7	7	7	2024-12-21	2024-12-27	650.00
8	8	8	2024-12-22	2024-12-26	800.00
9	9	9	2024-12-23	2024-12-26	260.00
10	10	10	2024-12-24	2024-12-30	1000.00
15	11	3	2024-12-14	2024-12-18	1200.00

4. 논의 사항

- What you learned while doing your homework (contents other than class hours),
- Describe difficulties during homework

Application Programming
- executeQuery(String : SQL): SELECT문을 수행할 때, return type : ResultSet
- executeUpdate(String: SQL): UPDATE, DELETE, INSERT문, return : Int(영향을 받은 로우의 수)
- execute(String: SQL) : SELECT문을 수행, Return type은 Boolean이며, getResultSet을 통해 SELECT 문의 수행결과를 가져올 수 있음
- ER 다이어그램은 개념적 설계의 결과물이다.
- 개념 설계를 표현하기 위해 개체-관계 모델(Entity-Relationship model)사용
- ER 다이어그램의 구성 요소
  - (Strong)Entity/Weak Entity
  - Relationship/Identifying Relationship
  - Attribute/Multivalued Attribute ex)전공→ 복수전공
  - composite attribute/derived attribute
- ER 다이어그램을 릴레이션으로 변환하기

1. Entity와 Weak Entity는 모두 릴레이션으로 변환하기
   - Weak Entity의 partial key는 owner Entity의 primary키와 결부시켜 사용한다.
2. RelationShip의 경우
   - 1 : 1 (total side에 외래키로 추가)
   - 1 : N (N-Side에 외래키로 추가)
   - N : M (Relation으로 만들기)
3. Mutivalued attribute는 릴레이션으로 표현
4. N-ary도 릴레이션으로 표현 (연결되어 있는 Entity들의 primary를 활용하여 릴레이션으로 표현)