โครงงานที่ 1 ระบบ Hotel Booking

6733053621 นางสาวโซฮารา มะหะหมัด

6733076021 นายตัรมีซี ดาโอะ

6733105621 นายธัญพิสิษฐ เนาวประดิษฐ์

6733227521 วรพงศ์ ทองขุนดำ

6733229821 วรภัทร จิตติชานนท์

6733282021 หัฏฐกร หัตถการ

รายงานฉบับนี้เป็นส่วนหนึ่งของรายวิชา 2110322 ระบบฐานข้อมูล (Database Systems)

ภาควิชาวิศวกรรมคอมพิวเตอร์ หลักสูตรวิศวกรรมคอมพิวเตอร์และเทคโนโลยีดิจิทัล

จุฬาลงกรณ์มหาวิทยาลัย ภาคเรียนที่ 2 ปีการศึกษา 2567

คำนำ

รายงานเล่มนี้จัดทำขึ้นเพื่อเป็นส่วนหนึ่งของรายวิชา 2110322 ระบบฐานข้อมูล (Database Systems) เพื่อศึกษาและรายงานเรื่องฐานข้อมูล ความสัมพันธ์ของข้อมูล การออกแบบโมเดลของข้อมูล โดยเฉพาะโมเดลข้อมูลเชิงความสัมพันธ์ (Relational Model Database) รวมถึงเทคนิคการออกแบบระบบฐานข้อมูล ประกอบด้วย ER Diagram for Relational Model, Schema refinement, SQL และ Document-based NoSQL ผ่านการทำโครงงานในครั้งนี้

ทางคณะผู้จัดทำหวังเป็นว่ารายงานฉบับนี้จะเป็นประโยชน์แก่ผู้อ่านที่กำลังศึกษาการออกแบบระบบและฐานข้อมูล หากมีข้อผิดพลาดประการใด คณะผู้จัดทำขอน้อมรับและขออภัยมา ณ ที่นี้

คณะผู้จัดทำ

สารบัญ

เรื่อง **หน้า**

บทนำ

โครงงาน “Hotel Booking” เป็นโครงงานด้านการออกแบบระบบฐานข้อมูล โดยมีพื้นฐานจากระบบการจองโรงแรม ซึ่งได้รับการพัฒนาโดยคำนึงถึงความต้องการของผู้ใช้งาน ระบบถูกออกแบบให้มีการทำงานได้อย่างมีประสิทธิภาพ รองรับการจัดเก็บข้อมูลของผู้ใช้งานที่สามารถจำแนกออกเป็น 3 ประเภท ได้แก่ ผู้ดูแลระบบ
ผู้ลงทะเบียน และ ผู้ไม่ได้ลงทะเบียน

ระบบสามารถบันทึกข้อมูลของโรงแรมและข้อมูลการจองโรงแรมได้อย่างเป็นระเบียบ ผู้ใช้งานสามารถแก้ไขข้อมูลการจองหรือข้อมูลของโรงแรมได้ตามสิทธิ์ของแต่ละประเภท นอกจากนี้ ระบบยังมีการบันทึกประวัติการทำรายการ เพื่อให้สามารถนำข้อมูลไปวิเคราะห์ ปรับปรุง และพัฒนาให้มีประสิทธิภาพมากยิ่งขึ้น การออกแบบระบบให้เป็นไปตามข้อจำกัดของ ระบบจัดการฐานข้อมูล (DBMS) ได้รับการพิจารณาอย่างรอบคอบเพื่อให้ข้อมูลมีความสัมพันธ์กันอย่างถูกต้อง คณะผู้จัดทำได้ออกแบบ ER Diagram ที่ประกอบด้วย 7 Entity Types และ 5 Relationships ก่อนนำไปพัฒนาและปรับปรุงจนได้ Schema Diagram ที่ประกอบด้วย 10 ตาราง (Tables) โดยผ่านกระบวนการ Normalization เพื่อลดความซ้ำซ้อนของข้อมูลและเพิ่มประสิทธิภาพในการใช้งาน

ER Diagram (Chen's Notation)

ในโครงงานระบบ Hotel Booking ได้มีการกำหนดรายละเอียดของ Entity types และ Relationship types ต่างๆ ไว้ดังต่อไปนี้

Entity types:

- USER เป็น Superclass แทนถึงผู้ใช้งานระบบ ซึ่งได้มีการ Specialization โดยแบ่ง Subclasses ออกเป็น ADMIN UNREGISTERED_USER และ REGISTERED USER
- HOTEL แทนถึงโรงแรมที่อยู่ในระบบ Hotel Booking
- ROOM เป็น Weak Entity Type ซึ่งมี Owner Entity คือ Hotel และ มี Identifying Relationship คือ Dependent
- BOOKING_TRANSACTION แทนถึง Booking ต่างๆ ที่เกิดขึ้นในระบบ

Relationship types:

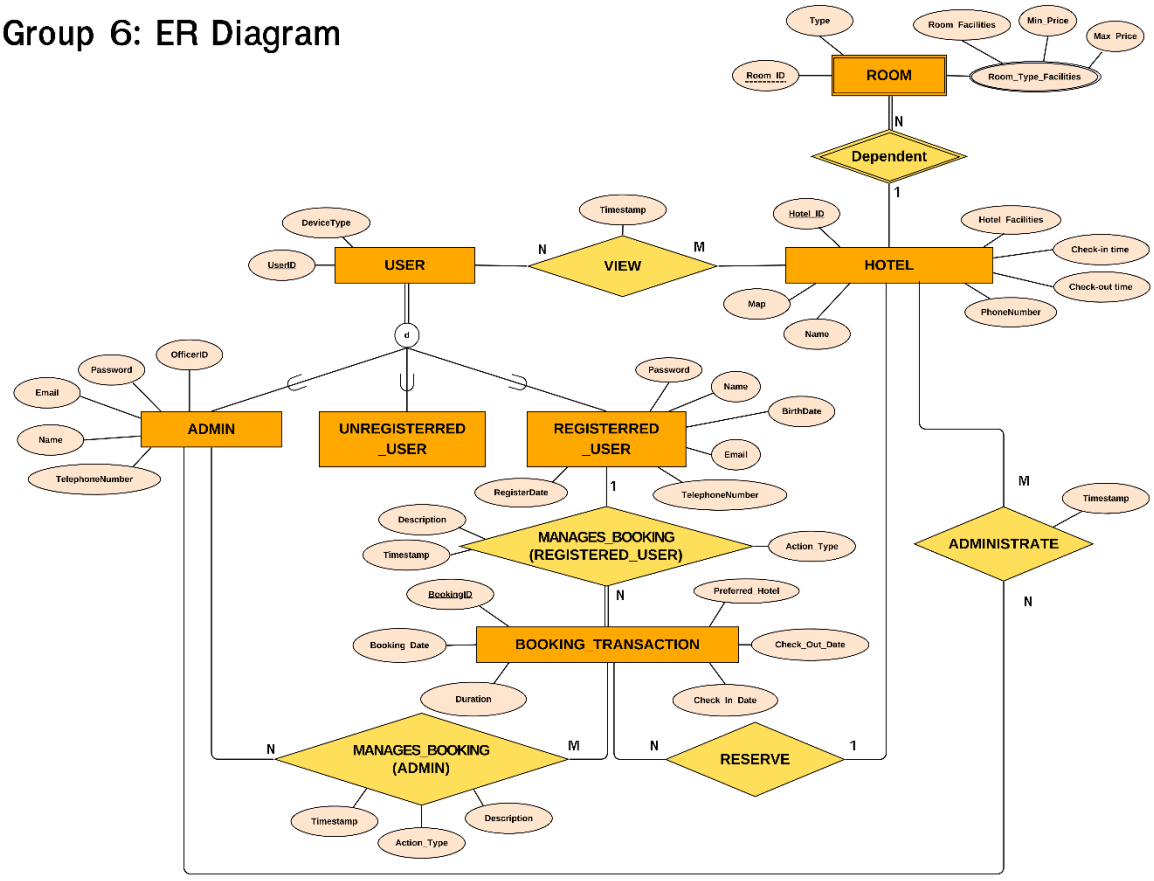
- VIEW เป็นความสัมพันธ์ระหว่าง USER และ HOTEL ซึ่งแสดงถึงความสามารถในการ view ข้อมูลโรงแรมที่อยู่ในระบบ
- MANAGE_BOOKING (ADMIN) เป็นความสัมพันธ์ของการกระทำที่เกิดระหว่าง ADMIN และ BOOKING_TRANSACTION (M:N) โดยการกระทำดังกล่าว ได้แก่ การ View Update Insert Delete Booking ของทุก USER
- MANAGE_BOOKING (REGISTERED_USER) เป็นความสัมพันธ์ของการกระทำที่เกิดระหว่าง REGISTERED_USER และ BOOKING_TRANSACTION (M:N) โดยการกระทำดังกล่าว ได้แก่ การ View Booking ของตนเอง
- RESERVE เป็นความสัมพันธ์ของการจองห้องที่เกิดระหว่าง BOOKING_TRANSACTION และ HOTEL (1:N)
- ADMINISTRATE เป็นความสัมพันธ์ของการกระทำที่เกิดระหว่าง ADMIN และ HOTEL (M:N) โดยการกระทำดังกล่าง ได้แก่ การ insert update delete HOTEL และการ insert update room และ room facilities type ของแต่ละ HOTEL

Attributes:

Entity Type	Attributes
Admin	UserID(PK), DeviceType, OfficerID, Password, Name, Email, TelephoneNumber
Unregistered_User	UserID(PK), DeviceType
Registered_User	UserID(PK), DeviceType, Password, Name, Email, TelephoneNumber, BirthDate, RegisterDate
Hotel	hotel_id (PK), map_url , hotel_name, hotel_phonenumber , PhoneNumber, check_in_time , check_out_time, hotel_facilities
Booking_Transaction	booking_id (PK), booking_date, duration, check_in_date, check_out_date, user_id, hotel_id, room_id
Room	room_id(Partial Key), room_type,room_type_facilities (min_price, max_price, room_facilities)

Relationship Type	Attributes
View	Timestamp
Administrate	Timestamp
Manage_Booking(Admin)	Timestamp, Description, Action_type
Manage_Booking(Registered_user)	Timestamp, Description, Action_type

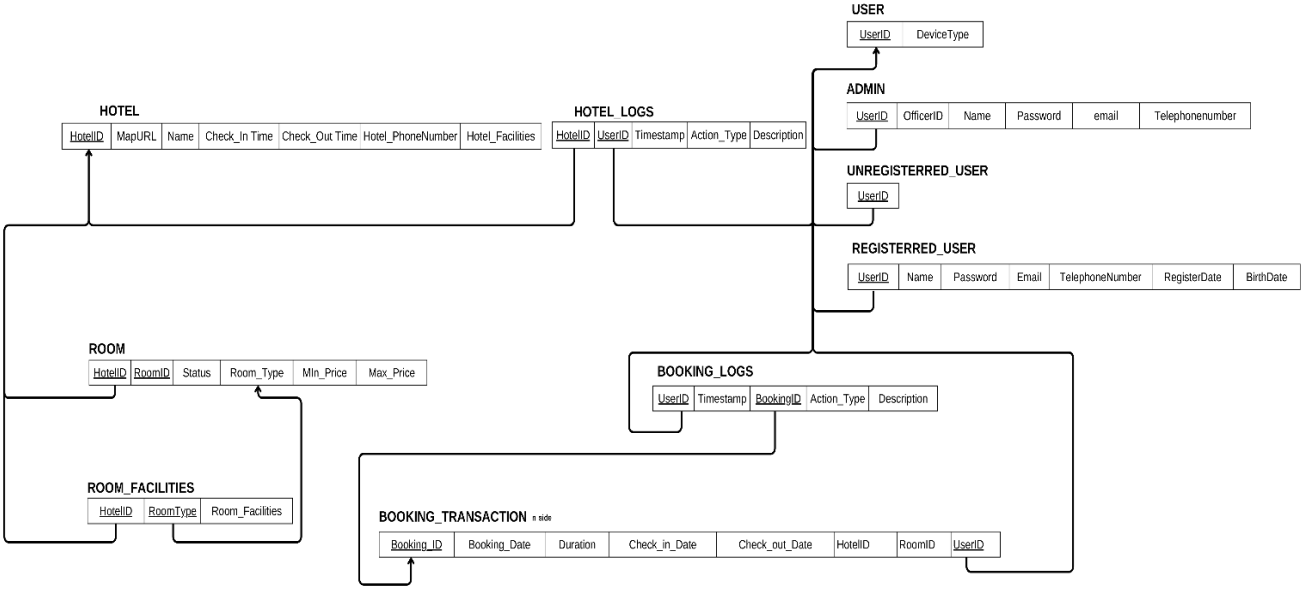
Group 6: ER Diagram



Schema Diagram

ในโครงงานระบบ Hotel Booking มีการจัดเก็บข้อมูลในรูปแบบของ Relational จึงมีโครงสร้างของ Schema ซึ่งใช้ Entities และ Relation จากการทำ ER-Diagram ในการออกแบบ นอกจากนี้ระบบฐานข้อมูลดังกล่าวได้มีการ ต่อยอดโดยการทำ Normalization เพื่อให้เหมาะกับการนำไปใช้สร้าง Database ในลำดับต่อไป โดยมีการกำหนดสร้าง Primary Key (PK), สำหรับในทุก ๆ Relation และ Foreign Key (FK) ในบาง Relation เพื่อเชื่อมโยง Relation ระหว่าง Entities

Entity Type	รายละเอียด	
User	Primary Key	user_id
	Foreign Key	-
Admin	Primary Key	-
	Foreign Key	user_id
Unregistered_User	Primary Key	-
	Foreign Key	user_id
Registered_User	Primary Key	-
	Foreign Key	user_id
Hotel	Primary Key	hotel_id
	Foreign Key	-
Room	Primary Key	hotel_id , room_id
	Foreign Key	hotel_id
Booking_Transaction	Primary Key	-
	Foreign Key	hotel_id , user_td , room_id
Hotel_Logs	Primary Key	-
	Foreign Key	user_id , hotel_id
Booking_logs	Primary Key	-
	Foreign Key	Booking_id , user_id
Room_Type_Facilities	Primary Key	-
	Foreign Key	hotel_id


USER
UserID
DeviceType
ADMIN
UserID
OfficerID
Name
Password
email
Telephonenumber
HOTEL
HotelID
MapURL
Name
Check_In Time
Check_Out Time
Hotel_PhoneNumber
Hotel_Facilities
HOTEL_LOGS
HotelID
UserID
Timestamp
Action_Type
Description
UNREGISTERRED_USER
UserID
REGISTERRED_USER
UserID
Name
Password
Email
TelephoneNumber
RegisterDate
BirthDate
ROOM
HotelID
RoomID
Status
Room_Type
MIn_Price
Max_Price
BOOKING_LOGS
UserID
Timestamp
BookingID
Action_Type
Description
ROOM_FACILITIES
HotelID
RoomType
Room_Facilities
BOOKING_TRANSACTION
n side
Booking_ID
Booking_Date
Duration
Check_in_Date
Check_out_Date
HotelID
RoomID
UserID

SQL Commands

1.) คำสั่งที่เกี่ยวกับ CREATE

1.1) สร้างตารางเก็บข้อมูลของผู้ใช้ทุกประเภท

คำอธิบาย: สร้าง table usert, admin, unregisterred_user และ registerred_user รวมทั้งสิ้น 4 tables

รันคำสั่งจาก: *SQL/create/create_user.sql*

คำสั่ง:

```
CREATE TABLE usert (
    user_id INT PRIMARY KEY,
    device_type VARCHAR(32) NOT NULL
);

CREATE TABLE admin(
    user_id INT,
    officer_id INT NOT NULL,
    admin_name VARCHAR(64) NOT NULL,
    admin_password VARCHAR(64) NOT NULL,
    admin_email VARCHAR(64) NOT NULL,
    telephone_number VARCHAR(10),
    FOREIGN KEY (user_id) REFERENCES usert(user_id)
);

CREATE TABLE unregisterred_user (
    user_id INT,
    FOREIGN KEY (user_id) REFERENCES usert(user_id)
);

CREATE TABLE registerred_user(
    user_id INT,
    user_name VARCHAR(64) NOT NULL,
    user_password VARCHAR(64) NOT NULL,
    user_email VARCHAR(64) NOT NULL,
    telephone_number VARCHAR(10),
    register_date TIMESTAMP NOT NULL,
    birth_date DATE NOT NULL,
    FOREIGN KEY(user_id) REFERENCES usert(user_id)
);
```

1.2) สร้างตารางสำหรับเก็บข้อมูลของโรงแรม

คำอธิบาย: สร้าง table hotel, room และ room_type_facilities รวมทั้งสิ้น 3 tables

รันคำสั่งจาก: SQL/create/create_hotel.sql

คำสั่ง:

```
CREATE TABLE hotel(
    hotel_id INT PRIMARY KEY,
    map_url VARCHAR DEFAULT NULL,
    hotel_name VARCHAR(256),
    hotel_phonenumber VARCHAR(10),
    check_in_time TIME,
    check_out_time TIME,
    hotel_facilities VARCHAR(512)
);

CREATE TABLE room(
```

```
    hotel_id INT,
    room_id INT,
    room_type VARCHAR(64),
    PRIMARY KEY(hotel_id, room_id),
    FOREIGN KEY(hotel_id) REFERENCES hotel(hotel_id)
);

CREATE TABLE room_type_facilities(
    hotel_id INT,
    room_type VARCHAR(64),
    min_price INT,
    max_price INT,
    room_facilities VARCHAR(256), --facilities description
    FOREIGN KEY(hotel_id) REFERENCES hotel(hotel_id)

);
```

1.3) สร้างตารางสำหรับเก็บคำสั่งจอง และ log ต่าง ๆ

คำอธิบาย: สร้าง table booking_transaction, hotel_log และ booking_log

รันคำสั่งจาก: *SQL/create/create_relationship.sql*

คำสั่ง:

```
CREATE TABLE booking_transaction(
    booking_id INT PRIMARY KEY,
    booking_date DATE NOT NULL,
    duration INT,
    check_in_date DATE NOT NULL,
    check_out_date DATE NOT NULL,
    user_id INT,
    hotel_id INT,
    room_id INT,
    FOREIGN KEY(hotel_id, room_id) REFERENCES room(hotel_id, room_id),
    FOREIGN KEY(user_id) REFERENCES usert(user_id)
);

CREATE TABLE hotel_log(
    user_id INT,
    hotel_id INT,
    action_type VARCHAR,
    action_timestamp TIMESTAMP,
    action_description VARCHAR(256),
    FOREIGN KEY (user_id) REFERENCES usert(user_id),
    FOREIGN KEY (hotel_id) REFERENCES hotel(hotel_id)
);

CREATE TABLE booking_log(
    user_id INT,
    booking_id INT,
    action_type VARCHAR,
    action_timestamp TIMESTAMP,
    action_description VARCHAR(256),
    FOREIGN KEY (booking_id) REFERENCES
        booking_transaction(booking_id),
    FOREIGN KEY (user_id) REFERENCES usert(user_id)
);
```

2.) คำสั่งที่เกี่ยวข้องกับการ INSERT

2.1) ใส่ข้อมูลตัวอย่างของ user

คำอธิบาย: สร้าง PROCEDURE insert_user ที่มี parameter 9 ตัว ได้แก่

p_device_type VARCHAR(32) คือ อุปกรณ์ที่ user ใช้

p_officer_id INT คือ รหัสพนักงาน

p_name VARCHAR(64) คือ ชื่อ user

p_password VARCHAR(64) คือ รหัสผ่าน

p_telephone_number VARCHAR(10) คือ หมายเลขโทรศัพท์ของ user

p_email VARCHAR(64) คือ email ของ user

p_register_date TIMESTAMP คือ เวลาลงทะเบียน

p_birth_date DATE คือ วันเกิด

p_user_type VARCHAR(32) คือ ประเภทของ user

โดย insert_user จะเพิ่มข้อมูลของ user ลงในตาราง usert และเพิ่มข้อมูลลงในตารางแยกตามประเภทของ user ได้แก่ admin, registerred_user และ unregisterred_user และจะมีการสร้าง user_id ด้วยฟังก์ชัน util_gen_user_id โดยใช้ข้อมูลตัวอย่างที่ประกอบด้วย admin 9 คน (user_id เริ่มที่ 900000000), registerred_user 63 คน (user_id เริ่มที่ 500000000) และ unregisterred_user 41 คน (user_id เริ่มที่ 100000000)

รันคำสั่งจาก:

1. *SQL/insert/util_gen_user_id.sql (สร้างฟังก์ชันสำหรับ generate user_id)*
2. *SQL/insert/insert_user.sql (สร้างฟังก์ชัน insert_user)*
3. *SQL/sample_data/sample_user.sql (ข้อมูลทดสอบ)*

คำสั่ง:

คำสั่งสร้างฟังก์ชัน generate id จากไฟล์ *SQL/insert/insert_user.sql*

```
CREATE OR REPLACE PROCEDURE insert_user(
    p_device_type VARCHAR(32),
    p_officer_id INT,
    p_name VARCHAR(64),
    p_passwORd VARCHAR(64),
    p_telephone_number VARCHAR(10),
    p_email VARCHAR(64),
    p_register_date TIMESTAMP,
    p_birth_date DATE,
    p_user_type VARCHAR(32)
)
LANGUAGE plpgsql
AS $$
DECLARE
    new_user_id INT;
BEGIN
    new_user_id := util_gen_user_id(p_user_type);
```

```
    IF p_user_type = 'registerred_user' THEN
        IF EXISTS (SELECT 1 FROM registerred_user WHERE user_email = p_email) THEN
            RAISE EXCEPTION 'Email already in use: %', p_email;
        END IF;
    ELSIF p_user_type = 'admin' THEN
        IF EXISTS (SELECT 1 FROM admin WHERE admin_email = p_email) THEN
            RAISE EXCEPTION 'Email already in use: %', p_email;
        END IF;
    END IF;

    INSERT INTO usert(user_id, device_type)
        VALUES (new_user_id, p_device_type);

    CASE p_user_type
        WHEN 'admin' THEN
            INSERT INTO admin (user_id, officer_id, admin_name, admin_passwORd,
telephone_number, admin_email)
                VALUES (new_user_id, p_officer_id, p_name, p_passwORd,
p_telephone_number, p_email);
        WHEN 'unregisterred_user' THEN
            INSERT INTO unregisterred_user (user_id)
                VALUES (new_user_id);
        WHEN 'registerred_user' THEN
            INSERT INTO registerred_user (user_id, user_name, user_passwORd,
user_email, telephone_number, register_date, birth_date)
                VALUES (new_user_id, p_name, p_passwORd, p_email,
p_telephone_number, p_register_date, p_birth_date);
        ELSE
            RAISE EXCEPTION 'Invalid user type: %', p_user_type;
    END CASE;
END; $$;
```

ตัวอย่างข้อมูลที่ INSERT เข้ามา

```
-- Example of admin
CALL insert_user('iOS', 113, 'Sittipong Wongsakul', 'password123', '1090123456',
'sittipong.wongsakul@example.com', CAST(NOW() AS TIMESTAMP), '1987-09-11', 'admin');

-- Example of registered user
CALL insert_user('iOS', NULL, 'Apichart Tanthanuch', 'regpass123', '1156789012',
'apichart.tanthanuch@example.com', CAST(NOW() AS TIMESTAMP), '1991-07-25',
'registerred_user');

-- Example of unregistered user
CALL insert_user('iOS', NULL, NULL, NULL, NULL, NULL, NULL, NULL,
'unregisterred_user');
```

ผลการรัน:

ผลที่ปรากฏใน usert (ยกตัวอย่างมา 5 ตัว)

	user_id [PK] integer	device_type character varying (32)
1	900000001	iOS
2	900000002	Android
3	900000003	Windows
4	500000001	iOS
5	500000002	Android

ผลที่ปรากฏใน unregisterred_user (ยกตัวอย่างมา 5 แถว)

	user_id integer
1	100000001
2	100000002
3	100000003
4	100000004
5	100000005

ผลที่ปรากฏใน admin (ยกตัวอย่างมา 5 แถว)

	user_id integer	officer_id integer	admin_name character varying (64)	admin_password character varying (64)	admin_email character varying (64)	telephone_number character varying (10)
1	900000001	113	Sittipong Wongsakul	password123	sittipong.wongsakul@example.com	1090123456
2	900000002	114	Pimchanok Jirawat	password456	pimchanok.jirawat@example.com	1101234567
3	900000003	115	Chanin Sutharak	password789	chanin.sutharak@example.com	1112345678
4	900000004	116	Thanakorn Rattanapong	password123	thanakorn.rattanapong@example.com	1189012345
5	900000005	117	Wassana Pongsuk	password456	wassana.pongsuk@example.com	1190123456

ผลที่ปรากฏใน registerred_user (ยกตัวอย่างมา 5 แถว)

	user_id integer	user_name character varying (64)	user_password character varying (64)	user_email character varying (64)	telephone_number character varying (10)	register_date timestamp without time zone	birth_date date
1	500000001	Sudarat Meesang	regpass123	sudarat.meesang@example.com	1123456789	2025-02-10 07:00:53.987227	1996-10-30
2	500000002	Sakchai Phongsak	regpass456	sakchai.phongsak@example.com	1134567890	2025-02-10 07:00:53.987227	1994-04-17
3	500000003	Thanyarat Rattanasak	regpass789	thanyarat.rattanasak@example.com	1145678901	2025-02-10 07:00:53.987227	1995-03-03
4	500000004	Apichart Tanthanuch	regpass123	apichart.tanthanuch@example.com	1156789012	2025-02-10 07:00:53.987227	1991-07-25
5	500000005	Kwanjai Chuchai	regpass456	kwanjai.chuchai@example.com	1167890123	2025-02-10 07:00:53.987227	1993-01-19

2.2) ใส่ข้อมูลตัวอย่างของ hotel

คำอธิบาย: สร้าง PROCEDURE insert_hotel ที่มี parameter 22 ตัว ได้แก่

admin_id INT คือ user_id ของ admin

map_url VARCHAR คือ Link แผนที่โรงแรม

hotel_name VARCHAR(256) คือ ชื่อโรงแรม

check_in TIME คือ เวลา Check-in ของโรงแรม

check_out TIME คือ เวลา Check-out ของโรงแรม

facilities VARCHAR(512) คือ สิ่งอำนวยความสะดวกของโรงแรม

hotel_number VARCHAR(10) คือ หมายเลขโทรศัพท์ของโรงแรม

num_floors INT คือ จำนวนชั้นของโรงแรม(ไม่นับชั้นที่ 1)

num_rooms_per_floor INT คือ จำนวนห้องใน 1 ชั้น

min1 INT คือ ราคาต่ำสุดของห้องราคาถูก

max1 INT คือ ราคาสูงสุดของห้องราคาถูก

min2 INT คือ ราคาต่ำสุดของห้องราคากลาง

max2 INT คือ ราคาสูงสุดของห้องราคากลาง

min3 INT คือ ราคาต่ำสุดของห้องราคาแพง

max3 INT คือ ราคาสูงสุดของห้องราคาแพง

c1 INT คือ ชั้นแรกที่เป็นห้องราคาถูก

c2 INT คือ ชั้นแรกที่เป็นห้องราคากลาง

c3 INT คือ ชั้นแรกที่เป็นห้องราคาแพง

facilities1 VARCHAR(256) คือ สิ่งอำนวยความสะดวกของห้องราคาถูก

facilities2 VARCHAR(256) คือ สิ่งอำนวยความสะดวกของห้องราคากลาง

facilities3 VARCHAR(256) คือ สิ่งอำนวยความสะดวกของห้องราคาแพง

insert_description VARCHAR(256) คือ comment จาก admin

โดย insert_hotel จะตรวจสอบ admin_id ที่รับเข้ามาเป็น admin จริงหรือไม่ ถ้าไม่ใช่
จะไม่สามารถเพิ่มข้อมูลโรงแรม ถ้าตรวจสอบแล้วพบว่าเป็น admin จริง จะเพิ่มข้อมูลของโรงแรมลงในตาราง hotel ตาม พารามิเตอร์ที่รับเข้ามา ใช้ฟังก์ชัน util_gen_hotel_id ในการสร้าง hotel_id และภายใน insert_hotel จะมีการเรียกใช้งาน insert_room_type_facilities
เพื่อกำหนดสิ่งอำนวยความสะดวกของห้องแต่ละประเภท ก่อนที่จะใช้งาน insert_room
เพื่อเพิ่มข้อมูลของห้องลงในตาราง room โดยกำหนด room_id ตามชั้นที่ห้องนั้นอยู่และลำดับห้อง เช่น room_id คือ 219 หมายถึง ห้องนี้เป็นห้องลำดับที่ 19 ของชั้นที่ 2 เมื่อเพิ่มข้อมูลห้องครบแล้ว จะเพิ่ม insert_description ลงในตาราง hotel_log ข้อมูลตัวอย่างที่ใช้ทดสอบประกอบด้วย ข้อมูลโรงแรมจำนวน 10 โรงแรม

รันคำสั่งจาก:

1. *SQL/insert/util_gen_hotel_id.sql (สร้างฟังก์ชัน generate id)*
2. *SQL/insert/insert_room_type_facilities.sql*
3. *SQL/insert/insert_room.sql*
4. *SQL/insert/insert_hotel.sql*
5. *SQL/sample_data/sample_hotel.sql (ข้อมูลทดสอบ)*

คำสั่ง:

ฟังก์ชันสำหรับการสร้าง hotel จากไฟล์ *SQL/insert/insert_hotel.sql*

```
CREATE OR REPLACE PROCEDURE insert_hotel(
    admin_id INT,
```

```
    map_url VARCHAR DEFAULT NULL,
    hotel_name VARCHAR(256) DEFAULT NULL,
    check_in TIME DEFAULT NULL,
    check_out TIME DEFAULT NULL,
    facilities VARCHAR(512) DEFAULT NULL,
    hotel_number VARCHAR(10) DEFAULT NULL,
    num_floors INT DEFAULT 5,          -- Number of floors of the hotel
    num_rooms_per_floor INT DEFAULT 20, -- Number of rooms per floor
    min1 INT DEFAULT 500,     -- Minimum price of a cheap room
    max1 INT DEFAULT 1000,    -- Maximum price of a cheap room
    min2 INT DEFAULT 1500,    -- Minimum price for a middle class room
    max2 INT DEFAULT 2000,    -- Highest price for a middle class room
    min3 INT DEFAULT 2500,    -- Minimum price of expensive rooms
    max3 INT DEFAULT 3000,    -- Maximum price of an expensive room
    c1 INT DEFAULT 2,         -- The first floor is a cheap room.
    c2 INT DEFAULT 5,         -- The first floor is a middle level room.
    c3 INT DEFAULT 8,         -- The first floor is an expensive room.
    facilities1 VARCHAR(256) DEFAULT NULL,
    facilities2 VARCHAR(256) DEFAULT NULL,
    facilities3 VARCHAR(256) DEFAULT NULL,
    insert_description VARCHAR(256) DEFAULT NULL
)
LANGUAGE plpgsql
AS $$
DECLARE
    is_admin INT;
    hotel_id INT;
    floor INT;
    room_id INT;
    room_facilities VARCHAR(256);
    room_type VARCHAR(64);
BEGIN
    hotel_id := util_gen_hotel_id();

    INSERT INTO hotel(hotel_id, map_url, hotel_name, check_in_time,
check_out_time, hotel_phonenumber, hotel_facilities)
        VALUES (hotel_id, map_url, hotel_name, check_in, check_out,
hotel_number, facilities);

    CALL insert_room_type_facilities(hotel_id, 'Cheap', facilities1,
min1, max1);
    CALL insert_room_type_facilities(hotel_id, 'Medium', facilities2,
min2, max2);
    CALL insert_room_type_facilities(hotel_id, 'Expensive', facilities3,
min3, max3);

    IF c1 < 2 OR c2 <= c1 OR c3 <= c2 OR c3 > (1 + num_floors) THEN
```

```
        RAISE EXCEPTION 'Invalid floor range: c1=%, c2=%, c3=% (Must be:
2 ≤ c1 < c2 < c3 ≤ num_floors + 1)', c1, c2, c3;
    END IF;

    FOR floor IN 2..(1 + num_floors) LOOP
        IF floor >= c1 AND floor < c2 THEN
            room_type := 'Cheap';
        ELSIF floor >= c2 AND floor < c3 THEN
            room_type := 'Medium';
        ELSE
            room_type := 'Expensive';
        END IF;
        FOR room_id IN 1..num_rooms_per_floor LOOP
            CALL insert_room(hotel_id, floor, (floor * 100) + room_id,
room_type);
        END LOOP;
    END LOOP;

    INSERT INTO hotel_log (hotel_id, user_id, action_timestamp,
action_type, action_description)
        VALUES (
            hotel_id,
            admin_id,
            CAST(NOW() AS TIMESTAMP),
            'admin_insert_hotel',
            insert_description
        );

    RAISE NOTICE 'Hotel added successfully with ID: %', hotel_id;
END; $$;
```

ตัวอย่างข้อมูลที่ INSERT เข้ามา

```
CALL insert_hotel(
    900000007,
    'http://example.com/hotel1',
    'Grand Palace Hotel',
    '14:00', '12:00',
    'WiFi, Pool, Valet Parking, Gym, Spa, Restaurant, Bar, Conference Room',
'0812345678',
    8, 25,
    800, 1500,  1800, 2500,  3000, 4000,
    2, 5, 7,
    'Single bed, ceiling fan, shared bathroom, free WiFi, basic toiletries, no
minibar',
    'Queen bed, air conditioning, private bathroom, TV, WiFi, minibar, toiletries',
    'King bed, air conditioning, private jacuzzi, smart TV, high-speed WiFi, luxury
toiletries, balcony, 24/7 room service'
);
```

ผลการรัน:

ผลที่ปรากฏใน hotel (ยกตัวอย่างมา 5 แถว)

	hotel_id [PK] integer	map_url character varying	hotel_name character varying (256)	hotel_phonenumber character varying (10)	check_in_time time without time zone	check_out_time time without time zone	hotel_facilities character varying (512)
1	10000	http://example.com/hotel1	Grand Palace Hotel	0812345678	14:00:00	12:00:00	WiFi, Pool, Valet Parking, Gym, Spa, Restaurant, Bar, Conference Room
2	10001	[null]	Cozy Cabin Inn	[null]	[null]	[null]	WiFi, Free Breakfast
3	10002	[null]	Beachfront Resort	0888888888	15:30:00	11:00:00	WiFi, Beach Access, Restaurant, Spa
4	10003	http://example.com/hotel4	Mountain View Lodge	0999999999	13:00:00	10:30:00	WiFi, Hiking Trails, Free Breakfast
5	10004	http://example.com/latecheckin	Downtown Plaza Hotel	0777777777	18:00:00	08:00:00	WiFi, Pool, Valet Parking, Gym, Spa, 24-hour Reception

ผลที่ปรากฏใน room_type_facilities (ยกตัวอย่างมา 6 แถว)

	hotel_id integer	room_type character varying (64)	min_price integer	max_price integer	room_facilities character varying (256)
1	10000	Cheap	800	1500	Single bed, ceiling fan, shared bathroom, free WiFi, basic toiletries, no minibar
2	10000	Medium	1800	2500	Queen bed, air conditioning, private bathroom, TV, WiFi, minibar, toiletries
3	10000	Expensive	3000	4000	King bed, air conditioning, private jacuzzi, smart TV, high-speed WiFi, luxury toiletries, balcony, 24/7 room service
4	10001	Cheap	200	300	Twin bed, heater, shared bathroom, free WiFi, basic toiletries, no TV
5	10001	Medium	400	500	Queen bed, heating, private bathroom, TV, WiFi, minibar, toiletries
6	10001	Expensive	600	700	King bed, fireplace, private jacuzzi, smart TV, high-speed WiFi, luxury toiletries, scenic balcony, 24/7 room service

ผลที่ปรากฏใน room (ยกตัวอย่างมา 5 แถว)

	hotel_id [PK] integer	room_id [PK] integer	room_type character varying (64)
262	10002	212	Cheap
263	10002	213	Cheap
264	10002	214	Cheap
265	10002	215	Cheap
266	10002	301	Medium
267	10002	302	Medium

ผลที่ปรากฏใน hotel_log (ยกตัวอย่างมา 5 ตัว)

	user_id integer	hotel_id integer	action_type character varying	action_timestamp timestamp without time zone	action_description character varying (256)
1	900000007	10000	admin_insert_hotel	2025-02-10 07:11:37.844066	[null]
2	900000008	10001	admin_insert_hotel	2025-02-10 07:11:37.844066	[null]
3	900000009	10002	admin_insert_hotel	2025-02-10 07:11:37.844066	[null]
4	900000009	10003	admin_insert_hotel	2025-02-10 07:11:37.844066	[null]
5	900000004	10004	admin_insert_hotel	2025-02-10 07:11:37.844066	[null]

2.3) ใส่ข้อมูลตัวอย่างการ booking

คำอธิบาย: สร้าง PROCEDURE book ที่มี parameter 6 ตัว ได้แก่

book_check_in_date DATE คือ วันที่ user check-in

book_check_out_date DATE คือ วันที่ user check-out

book_user_id INT คือ user_id ของผู้ที่จองโรงแรม

book_hotel_id INT คือ hotel_id ของโรงแรมที่ถูกจอง

book_room_id INTคือ room_id ของห้องที่ถูกจอง

book_description VARCHAR(256) คือ ข้อมูลที่บันทึกใน bookig_log

โดย book จะตรวจสอบระยะเวลาที่ user จองโรงแรม ถ้าจองเป็นระยะเวลามากกว่า 3 วัน จะไม่สามารถจองได้ ถ้าไม่เกิน 3 วัน จะตรวจสอบว่าในช่วงเวลานั้นมีคนอื่นจองห้องนั้นอยู่แล้วหรือไม่ ถ้าไม่มีจะเพิ่มข้อมูลการจองโรงแรมลงในตาราง booking_transaction ตามพารามิเตอร์ที่รับเข้ามา และใช้ util_gen_booking_id เพื่อสร้าง booking_id สำหรับการจองทุกครั้ง และเพิ่มข้อมูล book_description ลงในตาราง booking_log ข้อมูลตัวอย่างที่ใช้ทดสอบประกอบด้วย ข้อมูลการจองโรงแรมจำนวน 21 รายการ โดยเป็นของ admin จำนวน 10 รายการ และเป็นของ registerred_user จำนวน 11 รายการ (unregisterred_user ไม่สามารถจองโรงแรมได้)

รันคำสั่งจาก:

1. *SQL/ insert/util_gen_booking_id.sql (สร้างฟังก์ชัน generate id)*
2. *SQL/ insert/insert_booking.sql*
3. *SQL/sample_data/sample_booking.sql (ข้อมูลทดสอบ)*

คำสั่ง:

ฟังก์ชันสำหรับเพิ่มข้อมูลการจอง จากไฟล์ SQL/ insert/insert_booking.sql

```sql
CREATE OR REPLACE PROCEDURE book(
    book_check_in_date DATE,
    book_check_out_date DATE,
    book_user_id INT,
    book_hotel_id INT,
    book_room_id INT,
    book_description VARCHAR(256) DEFAULT NULL
)
LANGUAGE plpgsql
AS $$
DECLARE
    new_booking_id INT;
    duration INT;
    room_available BOOLEAN;
BEGIN
    new_booking_id = util_gen_booking_id();
    duration = book_check_out_date - book_check_in_date;

    -- If booking exceeds 3 nights, reject
    IF duration > 3 THEN
        RAISE EXCEPTION 'Booking denied: More than 3 nights. Duration: %', duration;

    -- If room is already booked for the requested dates
    ELSIF EXISTS (
        SELECT *
        FROM booking_transaction b
        WHERE b.room_id = book_room_id
          AND b.hotel_id = book_hotel_id
          AND (b.check_in_date, b.check_out_date)
              OVERLAPS (book_check_in_date, book_check_out_date)
    ) THEN
        RAISE EXCEPTION 'Room is already booked for the requested dates.';
```

```
    ELSE
        -- Add booking_transaction
        INSERT INTO booking_transaction (booking_id, booking_date, duration,
check_in_date, check_out_date, user_id, hotel_id, room_id)
        VALUES (new_booking_id, NOW(), duration, book_check_in_date,
book_check_out_date, book_user_id, book_hotel_id, book_room_id);

        -- Add log
        INSERT INTO booking_log (booking_id, user_id, action_timestamp, action_type,
action_description)
        VALUES (
            new_booking_id,
            book_user_id,
            CAST(NOW() AS TIMESTAMP),
            'user_insert_book',
            book_description
        );
    END IF;
END; $$;
```

ผลที่ปรากฏใน booking_transaction

	booking_id [PK] integer	booking_date date	duration integer	check_in_date date	check_out_date date	user_id integer	hotel_id integer	room_id integer
1	100000001	2025-02-10	2	2025-02-10	2025-02-12	900000001	10000	201
2	100000002	2025-02-10	2	2025-02-10	2025-02-12	900000001	10001	202
3	100000003	2025-02-10	2	2025-02-10	2025-02-12	900000001	10002	203
4	100000004	2025-02-10	2	2025-02-10	2025-02-12	900000001	10003	201
5	100000005	2025-02-10	2	2025-02-10	2025-02-12	900000001	10004	202

ผลที่ปรากฏใน booking_log

	user_id integer	booking_id integer	action_type character varying	action_timestamp timestamp without time zone	action_description character varying (256)
9	900000001	100000009	user_insert_book	2025-02-10 07:18:39.237044	[null]
10	900000001	100000010	user_insert_book	2025-02-10 07:18:39.237044	[null]
11	500000009	100000011	user_insert_book	2025-02-10 07:18:39.237044	[null]
12	500000002	100000012	user_insert_book	2025-02-10 07:18:39.237044	[null]
13	500000003	100000013	user_insert_book	2025-02-10 07:18:39.237044	[null]

3.) คำสั่งที่เกี่ยวข้องกับการ VIEW

3.1) ดูข้อมูลการจอง

คำอธิบาย: สร้างฟังก์ชันซึ่งรับพารามิเตอร์ 3 ตัว ได้แก่

admin_status คือสถานะการเป็นแอดมิน

action_type คือฟิลเตอร์ที่จะใช้ในการกรองการดู

search_value คือค่าที่ต้องการดู ซึ่งค่านี้จะต้องสอดคล้องกับ action_type

ฟังก์ชัน view booking แบ่งการทำงานออกเป็น 2 กรณีคือ กรณีที่ admin_status = false ซึ่งหมายความว่าผู้เรียกใช้ฟังก์ชันคือ Unregisterred_User และกรณีที่ admin_status = true ซึ่งหมายความว่าผู้เรียกใช้ฟังก์ชันคือ Admin

ในกรณีแรกผู้เรียกใช้ฟังก์ชันสามารถดูข้อมูลการจแงของตนเองเท่านั้นผ่านการส่งพารามิเตอร์ booking_id ของตน ส่วนในกรณีที่สองผู้เรียกใช้ฟังก์ชันสามารถดูข้อมูลการจองของทุกคนตามการกำหนด action_type และ search_value ที่ต้องการ

รันคำสั่งจาก:

1. *SQL/view/view_booking.sql*
2. *SQL/sample_data/sample_view_booking.sql (ข้อมูลทดสอบ)*

คำสั่ง:

ฟังก์ชันสำหรับการดูข้อมูลการจอง จากไฟล์ *SQL\view\view_booking.sql*

```
CREATE OR REPLACE FUNCTION view_booking(
    admin_status BOOLEAN,
    action_type VARCHAR,
    search_value VARCHAR
)
RETURNS TABLE (
    booking_id INTEGER,
    booking_date DATE,
    check_in_date DATE,
    check_out_date DATE,
    duration INT,
    user_id INT,
    hotel_id INT,
    room_id INT
)
LANGUAGE plpgsql
AS $$
BEGIN
    -- Case 1: Not admin, only allows searching by booking ID (validate first)
    IF admin_status = FALSE THEN
        IF search_value ~ '^[0-9]+$' THEN
            RETURN QUERY
            SELECT b.booking_id, b.booking_date::DATE, b.check_in_date,
b.check_out_date, b.duration, b.user_id, b.hotel_id, b.room_id
            FROM booking_transaction b
```

```
            WHERE b.booking_id = search_value::INT;
        ELSE
            RAISE EXCEPTION 'Invalid search value for non-admin: %', search_value;
        END IF;

    -- Case 2: Admin - Select all rows when Action is "None" and Search Value is "-"
    ELSIF admin_status = TRUE AND action_type = 'None' AND search_value = '-' THEN
        RETURN QUERY
        SELECT b.booking_id, b.booking_date::DATE, b.check_in_date,
b.check_out_date, b.duration, b.user_id, b.hotel_id, b.room_id
        FROM booking_transaction b
        ORDER BY b.booking_date DESC;

    -- Case 3: Admin - Filter by booking ID
    ELSIF admin_status = TRUE AND action_type = 'booking ID' AND search_value ~
'^[0-9]+$' THEN
        RETURN QUERY
        SELECT b.booking_id, b.booking_date::DATE, b.check_in_date,
b.check_out_date, b.duration, b.user_id, b.hotel_id, b.room_id
        FROM booking_transaction b
        WHERE b.booking_id = search_value::INT;

    -- Case 4: Admin - Filter by duration
    ELSIF admin_status = TRUE AND action_type = 'duration' AND search_value ~ '^[0-
9]+$' THEN
        RETURN QUERY
        SELECT b.booking_id, b.booking_date::DATE, b.check_in_date,
b.check_out_date, b.duration, b.user_id, b.hotel_id, b.room_id
        FROM booking_transaction b
        WHERE b.duration = search_value::INT;

    -- Case 5: Admin - Filter by check-in date
    ELSIF admin_status = TRUE AND action_type = 'check_in_date' AND search_value ~
'^\d{4}-\d{2}-\d{2}$' THEN
        RETURN QUERY
        SELECT b.booking_id, b.booking_date::DATE, b.check_in_date,
b.check_out_date, b.duration, b.user_id, b.hotel_id, b.room_id
        FROM booking_transaction b
        WHERE b.check_in_date = search_value::DATE;

    -- Case 6: Admin - Filter by check-out date
    ELSIF admin_status = TRUE AND action_type = 'check_out_date' AND search_value ~
'^\d{4}-\d{2}-\d{2}$' THEN
        RETURN QUERY
        SELECT b.booking_id, b.booking_date::DATE, b.check_in_date,
b.check_out_date, b.duration, b.user_id, b.hotel_id, b.room_id
        FROM booking_transaction b
        WHERE b.check_out_date = search_value::DATE;

    -- Case 7: Admin - Filter by booking date (Ensure it's properly handled)
    ELSIF admin_status = TRUE AND action_type = 'booking_date' AND search_value ~
'^\d{4}-\d{2}-\d{2}$' THEN
        RETURN QUERY
        SELECT b.booking_id, b.booking_date::DATE, b.check_in_date,
b.check_out_date, b.duration, b.user_id, b.hotel_id, b.room_id
```

```
        FROM booking_transaction b
        WHERE b.booking_date::DATE = search_value::DATE;

    -- Case 8: Admin - Filter by user ID
    ELSIF admin_status = TRUE AND action_type = 'user_id' AND search_value ~ '^[0-
9]+$' THEN
        RETURN QUERY
        SELECT b.booking_id, b.booking_date::DATE, b.check_in_date,
b.check_out_date, b.duration, b.user_id, b.hotel_id, b.room_id
        FROM booking_transaction b
        WHERE b.user_id = search_value::INT;

    -- Case 9: Admin - Filter by hotel ID
    ELSIF admin_status = TRUE AND action_type = 'hotel_id' AND search_value ~ '^[0-
9]+$' THEN
        RETURN QUERY
        SELECT b.booking_id, b.booking_date::DATE, b.check_in_date,
b.check_out_date, b.duration, b.user_id, b.hotel_id, b.room_id
        FROM booking_transaction b
        WHERE b.hotel_id = search_value::INT;

    -- Case 10: Admin - Filter by room ID
    ELSIF admin_status = TRUE AND action_type = 'room_id' AND search_value ~ '^[0-
9]+$' THEN
        RETURN QUERY
        SELECT b.booking_id, b.booking_date::DATE, b.check_in_date,
b.check_out_date, b.duration, b.user_id, b.hotel_id, b.room_id
        FROM booking_transaction b
        WHERE b.room_id = search_value::INT;

    ELSE
        RAISE EXCEPTION 'Invalid action type or search value: % - %', action_type,
search_value;
    END IF;
END;
$$
```

ผลการรัน

กรณีที่ 1) ถ้าไม่ใช่ admin สามาถค้นหาได้ด้วย booking_id

```
SELECT * FROM view_booking(False,NULL,'100000001');
```

	booking_id integer	booking_date date	check_in_date date	check_out_date date	duration integer	user_id integer	hotel_id integer	room_id integer
1	100000021	2025-02-10	2025-02-10	2025-02-13	3	500000004	10009	309

กรณีที่ 2) admin ดูข้อมูลการจองทั้งหมด (ยกตัวอย่างมา 5 แถว)

```
SELECT * FROM view_booking(True,'None','-');
```

	booking_id integer	booking_date date	check_in_date date	check_out_date date	duration integer	user_id integer	hotel_id integer	room_id integer
1	100000001	2025-02-10	2025-02-10	2025-02-12	2	900000001	10000	201
2	100000002	2025-02-10	2025-02-10	2025-02-12	2	900000001	10001	202
3	100000003	2025-02-10	2025-02-10	2025-02-12	2	900000001	10002	203
4	100000004	2025-02-10	2025-02-10	2025-02-12	2	900000001	10003	201
5	100000005	2025-02-10	2025-02-10	2025-02-12	2	900000001	10004	202

กรณีที่ 3) admin ค้นหาด้วย booking_id

```
SELECT * FROM view_booking(True,'booking ID','100000019');
```

	booking_id integer	booking_date date	check_in_date date	check_out_date date	duration integer	user_id integer	hotel_id integer	room_id integer
1	100000019	2025-02-10	2025-02-10	2025-02-12	2	500000012	10008	303

กรณีที่ 4) admin ค้นหาด้วย duration (ยกตัวอย่างมา 5 แถว)

```
SELECT * FROM view_booking(True,'duration','2');
```

	booking_id integer	booking_date date	check_in_date date	check_out_date date	duration integer	user_id integer	hotel_id integer	room_id integer
1	100000001	2025-02-10	2025-02-10	2025-02-12	2	900000001	10000	201
2	100000002	2025-02-10	2025-02-10	2025-02-12	2	900000001	10001	202
3	100000003	2025-02-10	2025-02-10	2025-02-12	2	900000001	10002	203
4	100000004	2025-02-10	2025-02-10	2025-02-12	2	900000001	10003	201
5	100000005	2025-02-10	2025-02-10	2025-02-12	2	900000001	10004	202

กรณีที่ 5) admin ค้นหาด้วย check_in_date (ยกตัวอย่างมา 5 แถว)

```
SELECT * FROM view_booking(True,'check_in_date','2025-02-10');
```

	booking_id integer	booking_date date	check_in_date date	check_out_date date	duration integer	user_id integer	hotel_id integer	room_id integer
1	100000001	2025-02-10	2025-02-10	2025-02-12	2	900000001	10000	201
2	100000002	2025-02-10	2025-02-10	2025-02-12	2	900000001	10001	202
3	100000003	2025-02-10	2025-02-10	2025-02-12	2	900000001	10002	203
4	100000004	2025-02-10	2025-02-10	2025-02-12	2	900000001	10003	201
5	100000005	2025-02-10	2025-02-10	2025-02-12	2	900000001	10004	202

กรณีที่ 6) admin ค้นหาด้วย check_out_date (ยกตัวอย่างมา 5 แถว)

```
SELECT * FROM view_booking(True,'check_out_date','2025-02-12');
```

	booking_id integer	booking_date date	check_in_date date	check_out_date date	duration integer	user_id integer	hotel_id integer	room_id integer
1	100000001	2025-02-10	2025-02-10	2025-02-12	2	900000001	10000	201
2	100000002	2025-02-10	2025-02-10	2025-02-12	2	900000001	10001	202
3	100000003	2025-02-10	2025-02-10	2025-02-12	2	900000001	10002	203
4	100000004	2025-02-10	2025-02-10	2025-02-12	2	900000001	10003	201
5	100000005	2025-02-10	2025-02-10	2025-02-12	2	900000001	10004	202

กรณีที่ 7) admin ค้นหาด้วย booking_date (ยกตัวอย่างมา 5 แถว)

```
SELECT * FROM view_booking(True,'booking_date','2025-02-10');
```

	booking_id integer	booking_date date	check_in_date date	check_out_date date	duration integer	user_id integer	hotel_id integer	room_id integer
1	100000001	2025-02-10	2025-02-10	2025-02-12	2	900000001	10000	201
2	100000002	2025-02-10	2025-02-10	2025-02-12	2	900000001	10001	202
3	100000003	2025-02-10	2025-02-10	2025-02-12	2	900000001	10002	203
4	100000004	2025-02-10	2025-02-10	2025-02-12	2	900000001	10003	201
5	100000005	2025-02-10	2025-02-10	2025-02-12	2	900000001	10004	202

กรณีที่ 8) admin ค้นหาด้วย user_id

```
SELECT * FROM view_booking(True,'user_id','500000002');
```

	booking_id integer	booking_date date	check_in_date date	check_out_date date	duration integer	user_id integer	hotel_id integer	room_id integer
1	100000012	2025-02-10	2025-02-10	2025-02-12	2	500000002	10001	302

กรณีที่ 9) admin ค้นหาด้วย hotel_id

```
SELECT * FROM view_booking(True,'hotel_id','10001');
```

	booking_id integer	booking_date date	check_in_date date	check_out_date date	duration integer	user_id integer	hotel_id integer	room_id integer
1	100000002	2025-02-10	2025-02-10	2025-02-12	2	900000001	10001	202
2	100000012	2025-02-10	2025-02-10	2025-02-12	2	500000002	10001	302

กรณีที่ 10) admin ค้นหาด้วย room_id

```
SELECT * FROM view_booking(True,'room_id','201');
```

	booking_id integer	booking_date date	check_in_date date	check_out_date date	duration integer	user_id integer	hotel_id integer	room_id integer
1	100000001	2025-02-10	2025-02-10	2025-02-12	2	900000001	10000	201
2	100000004	2025-02-10	2025-02-10	2025-02-12	2	900000001	10003	201
3	100000007	2025-02-10	2025-02-10	2025-02-12	2	900000001	10006	201
4	100000010	2025-02-10	2025-02-10	2025-02-12	2	900000001	10009	201

3.2) ดูข้อมูลโรงแรม

คำอธิบาย: คือฟังก์ชันการขอดูรายละเอียดของโรงแรมใดโรงแรมหนึ่งโดยส่งพารามิเตอร์ 2 ค่า ได้แก่

view_hotel_id คือไอดีของโรงแรมที่ต้องการดูรายละเอียด

view_user_id คือไอดีของ user ที่ขอดูรายละเอียด

รันคำสั่งจาก:

1. *SQL/view/view_hotel.sql*
2. *SQL/sample_data/sample_view_hotel.sql* (ข้อมูลทดสอบ)

คำสั่ง:

ฟังก์ชันสำหรับการดูข้อมูลโรงแรม จากไฟล์ *SQL/view/view_hotel.sql*

```
CREATE OR REPLACE FUNCTION view_hotel(
    view_hotel_id INT,
    view_user_id INT
)
```

```
RETURNS TABLE(
    hotel_id INT,
    hotel_name VARCHAR,
    map_url VARCHAR,
    hotel_phonenumber VARCHAR,
    check_in_time TIME,
    check_out_time TIME,
    hotel_facilities VARCHAR,
    available_room INT,
    room_type VARCHAR,
    min_price INT,
    max_price INT,
    room_facilities VARCHAR
)
LANGUAGE plpgsql
AS $$
DECLARE
    available INT;
BEGIN
    IF EXISTS (
        SELECT 1 FROM hotel h WHERE h.hotel_id = view_hotel_id
    ) THEN
        IF view_user_id < 900000000 THEN
            INSERT INTO hotel_log(hotel_id, user_id, action_timestamp,
action_type, action_description)
            VALUES (view_hotel_id, view_user_id, NOW(),
'user_view_hotel', NULL);
        END IF;

        SELECT COUNT(*) INTO available FROM room r WHERE r.hotel_id =
view_hotel_id ;
        RETURN QUERY
        SELECT h.hotel_id, h.hotel_name, h.map_url, h.hotel_phonenumber,
               CAST(h.check_in_time AS TIME), CAST(h.check_out_time AS
TIME), h.hotel_facilities, available,
               r.room_type,
               MIN(rtf.min_price) AS min_price,
               MAX(rtf.max_price) AS max_price,
               rtf.room_facilities
        FROM hotel h
        LEFT JOIN room r ON h.hotel_id = r.hotel_id
        LEFT JOIN room_type_facilities rtf ON r.hotel_id = rtf.hotel_id
AND r.room_type = rtf.room_type
        WHERE h.hotel_id = view_hotel_id
        GROUP BY h.hotel_id, h.hotel_name, h.map_url,
h.hotel_phonenumber,
```

```
                h.check_in_time, h.check_out_time, h.hotel_facilities,
available, r.room_type, rtf.room_facilities;
    ELSE
        RAISE NOTICE 'Hotel ID % does not exist', view_hotel_id;
    END IF;
END; $$;
```

ผลการรัน

	hotel_id integer	hotel_name character varying	map_url character varying	hotel_phonenumber character varying	check_in_time time without time zone	check_out_time time without time zone	hotel_facilities character varying	available_room integer	room_type character varying	min_price integer	max_price integer
1	10004	Downtown Plaza Hot...	http://example.com/latecheck...	0777777777	18:00:00	08:00:00	WiFi, Pool, Valet Parking, Gym, Spa, 24-hour Recepti...	198	Cheap	750	1300
2	10004	Downtown Plaza Hot...	http://example.com/latecheck...	0777777777	18:00:00	08:00:00	WiFi, Pool, Valet Parking, Gym, Spa, 24-hour Recepti...	198	Expensive	3000	3800
3	10004	Downtown Plaza Hot...	http://example.com/latecheck...	0777777777	18:00:00	08:00:00	WiFi, Pool, Valet Parking, Gym, Spa, 24-hour Recepti...	198	Medium	1800	2400

4.) คำสั่งที่เกี่ยวข้องกับการ UPDATE

4.1) แก้ไขข้อมูล booking

คำอธิบาย: เป็นคำสั่งที่ register user ที่ทำการจองหรือ admin ทำการแก้ไขข้อมูลการจอง ซึ่งสามารถแก้ไขได้แค่วันสำหรับ check in กับ check out

รันคำสั่งจาก:

1. *SQL/update/update_booking.sql*
2. *SQL/sample_data/sample_update_booking.sql* (ข้อมูลทดสอบ)

คำสั่ง:

คำสั่งสำหรับการแก้ไขข้อมูลการจอง *SQL/update/update_booking.sql*

```
CREATE OR REPLACE PROCEDURE update_booking(
    booking_id_update INT,
    update_check_in DATE,
    update_check_out DATE,
    user_id_update INT,
    book_room_id INT,
    book_hotel_id INT,
    description VARCHAR(256) DEFAULT NULL
)
LANGUAGE PLPGSQL
AS
$$
BEGIN
    -- first check if the user is not allow to edit
    IF NOT EXISTS (
        SELECT 1
        FROM admin a, registerred_user r
        WHERE a.user_id = user_id_update OR r.user_id = user_id_update
    ) THEN
        RAISE EXCEPTION 'there are no user id %', user_id_update;

    ELSE
    -- check that check in time must come before check out time
        IF update_check_in > update_check_out THEN
            RAISE EXCEPTION 'check in time must come before check out time';
        ELSIF EXISTS (
            SELECT *
            FROM booking_transaction b
            WHERE b.room_id = book_room_id
            AND b.hotel_id = book_hotel_id
            AND (
        -- Check if the new booking range overlaps with an existing one
                    (b.check_in_date, b.check_out_date)
                OVERLAPS (update_check_in, update_check_out)
            )
        ) THEN
            RAISE EXCEPTION 'Room is already booked for the requested dates.';
```

```
        ELSIF update_check_out - update_check_in <= 3 THEN -- check that duration
must within 3 nights
            -- update that booking transaction
            UPDATE booking_transaction
            SET check_in_date = update_check_in,
            check_out_date = update_check_out,
            duration = update_check_out - update_check_in
            WHERE booking_id = booking_id_update;

            -- insert log into booking_log
            INSERT INTO booking_log(user_id, booking_id, action_type,
action_timestamp, action_description)
            VALUES (
                user_id_update,
                booking_id_update,
                'user_update_booking',
                cast(NOW() AS TIMESTAMP),
                description
            );

            RAISE NOTICE 'change date successfully';

        ELSE
            RAISE EXCEPTION 'It more than 3 night';
        END IF;
    END IF;
END; $$
```

ก่อนแก้

	booking_id [PK] integer	booking_date date	duration integer	check_in_date date	check_out_date date	user_id integer	hotel_id integer	room_id integer
1	100000001	2025-02-10	2	2025-02-10	2025-02-12	900000001	10000	201
2	100000002	2025-02-10	2	2025-02-10	2025-02-12	900000001	10001	202
3	100000003	2025-02-10	2	2025-02-10	2025-02-12	900000001	10002	203
4	100000004	2025-02-10	2	2025-02-10	2025-02-12	900000001	10003	201
5	100000005	2025-02-10	2	2025-02-10	2025-02-12	900000001	10004	202

หลังแก้

	booking_id [PK] integer	booking_date date	duration integer	check_in_date date	check_out_date date	user_id integer	hotel_id integer	room_id integer
1	100000001	2025-02-10	2	2025-03-10	2025-03-12	900000001	10000	201
2	100000002	2025-02-10	2	2025-02-21	2025-02-23	900000001	10001	202
3	100000003	2025-02-10	1	2025-04-11	2025-04-12	900000001	10002	203
4	100000004	2025-02-10	2	2025-02-19	2025-02-21	900000001	10003	201
5	100000005	2025-02-10	2	2025-06-10	2025-06-12	900000001	10004	202

ผลที่ปรากฏใน booking_log

22	900000001	100000001	user_update_booki...	2025-02-10 08:08:50.932679	[null]
23	900000001	100000002	user_update_booki...	2025-02-10 08:08:56.597622	[null]
24	900000001	100000003	user_update_booki...	2025-02-10 08:09:01.457249	[null]
25	900000001	100000004	user_update_booki...	2025-02-10 08:09:06.478125	[null]
26	900000001	100000005	user_update_booki...	2025-02-10 08:09:11.477921	[null]

4.2) แก้ไขข้อมูล hotel

คำอธิบาย: คำสั่งที่ admin ทำการแก้ไขข้อมูลของโรงแรม

รันคำสั่งจาก:

1. *SQL/update/update_hotel.sql*
2. *SQL/sample_data/sample_update_hotel.sql (ข้อมูลทดสอบ)*

คำสั่ง:

คำสั่งแก้ไข้ข้อมูล hotel จากไฟล์ SQL/update/update_hotel.sql

```
CREATE OR REPLACE PROCEDURE update_hotel(
    p_hotel_id INT,
    admin_id INT,
    new_map_url VARCHAR DEFAULT NULL,
    new_hotel_name VARCHAR(256) DEFAULT NULL,
    new_check_in TIME DEFAULT NULL,
    new_check_out TIME DEFAULT NULL,
    new_hotel_facilities VARCHAR(512) DEFAULT NULL,
    new_hotel_phonenumber VARCHAR(10) DEFAULT NULL,
    insert_description VARCHAR(256) DEFAULT NULL
) AS $$
DECLARE
    is_admin INT;
    exists_hotel INT;
BEGIN
    SELECT COUNT(*) INTO is_admin FROM admin WHERE user_id = admin_id;
    IF is_admin = 0 THEN
        RAISE EXCEPTION 'Permission denied: Only admins can update hotels';
    END IF;

    SELECT COUNT(*) INTO exists_hotel FROM hotel h WHERE h.hotel_id = p_hotel_id;
    IF exists_hotel = 0 THEN
        RAISE EXCEPTION 'Error: Hotel does not exist';
    END IF;

    UPDATE hotel h
    SET map_url = COALESCE(new_map_url, map_url),
        hotel_name = COALESCE(new_hotel_name, hotel_name),
        check_in_time = COALESCE(new_check_in, check_in_time),
        check_out_time = COALESCE(new_check_out, check_out_time),
        hotel_phonenumber = COALESCE(new_hotel_phonenumber, hotel_phonenumber),
```

```
        hotel_facilities = COALESCE(new_hotel_facilities, hotel_facilities)
    WHERE h.hotel_id = p_hotel_id;

    INSERT INTO hotel_log (hotel_id, user_id, action_timestamp, action_type,
action_description)
        VALUES (
            p_hotel_id,
            admin_id,
            CAST(NOW() AS TIMESTAMP),
            'admin_update_hotel',
            insert_description
        );
    RAISE NOTICE 'Hotel updated successfully';
END;
$$ LANGUAGE plpgsql;
```

ก่อนแก้

	hotel_id [PK] integer	map_url character varying	hotel_name character varying (256)	hotel_phonenumber character varying (10)	check_in_time time without time zone	check_out_time time without time zone	hotel_facilities character varying (512)
1	10000	http://example.com/hotel1	Grand Palace Hotel	0812345678	14:00:00	12:00:00	WiFi, Pool, Valet Parking, Gym, Spa, Restaurant, Bar, Conference Room
2	10001	[null]	Cozy Cabin Inn	[null]	[null]	[null]	WiFi, Free Breakfast
3	10002	[null]	Beachfront Resort	0888888888	15:30:00	11:00:00	WiFi, Beach Access, Restaurant, Spa
4	10003	http://example.com/hotel4	Mountain View Lodge	0999999999	13:00:00	10:30:00	WiFi, Hiking Trails, Free Breakfast
5	10004	http://example.com/latecheckin	Downtown Plaza Hotel	0777777777	18:00:00	08:00:00	WiFi, Pool, Valet Parking, Gym, Spa, 24-hour Reception

หลังแก้

	hotel_id [PK] integer	map_url character varying	hotel_name character varying (256)	hotel_phonenumber character varying (10)	check_in_time time without time zone	check_out_time time without time zone	hotel_facilities character varying (512)
1	10000	http://example.com/max1	Super Skyscraper Hotel	0111111111	12:00:00	10:00:00	WiFi, Pool, Valet Parking, Rooftop Bar, Business Center, Gym, Spa, Indoor Pool
2	10001	http://example.com/max2	Luxury Haven Resort	0222222222	13:30:00	11:00:00	WiFi, Pool, Valet Parking, Butler Service, Private Beach, Fine Dining, Spa, Private Cabanas
3	10002	[null]	Beachfront Resort	0888888888	15:30:00	11:00:00	WiFi, Beach Access, Restaurant, Spa
4	10003	http://example.com/hotel4	Mountain View Lodge	0999999999	13:00:00	10:30:00	WiFi, Hiking Trails, Free Breakfast
5	10004	http://example.com/latecheckin	Beachfront Resort	0888888888	15:30:00	11:00:00	WiFi, Beach Access, Restaurant, Spa

ผลที่ปรากฏใน hotel_log

12	900000009	10003	admin_update_hot...	2025-02-10 08:14:23.846617	[null]
13	900000005	10000	admin_update_hot...	2025-02-10 08:14:23.846617	[null]
14	900000005	10001	admin_update_hot...	2025-02-10 08:14:23.846617	[null]
15	900000007	10007	admin_update_hot...	2025-02-10 08:14:23.846617	[null]
16	900000002	10005	admin_update_hot...	2025-02-10 08:14:23.846617	[null]

4.3) แก้ไข้ข้อมูล room

คำอธิบาย: คำสั่งที่ admin ทำการแก้ไขข้อมูลของ room type ของห้องภายในโรงแรม

รันคำสั่งจาก:

1. *SQL/update/update_room.sql*
2. *SQL/sample_data/sample_update_room.sql (ข้อมูลทดสอบ)*

คำสั่ง:

คำสั่งแก้ไขข้อมูลห้อง จากไฟล์ SQL/update/update_room.sql

```
CREATE OR REPLACE PROCEDURE update_room(
    update_hotel_id INT,
    update_room_id INT,
    update_room_type VARCHAR(64)
)
```

```
LANGUAGE plpgsql
AS $$
BEGIN
    UPDATE room
    SET room_type = update_room_type
        WHERE hotel_id = update_hotel_id AND room_id = update_room_id;
    RAISE NOTICE 'Room updated successfully';
END; $$;
```

ก่อนแก้

	hotel_id [PK] integer	room_id [PK] integer	room_type character varying (64)
1	10000	201	Cheap
2	10000	202	Cheap
3	10000	203	Cheap
4	10000	204	Cheap
5	10000	205	Cheap

หลังแก้

	hotel_id [PK] integer	room_id [PK] integer	room_type character varying (64)
1	10000	201	Premium
2	10000	202	Cheap
3	10000	203	Cheap
4	10000	204	Cheap
5	10000	205	Cheap

4.4) แก้ไขข้อมูล hotel facilities

คำอธิบาย: คำสั่งที่ admin แก้ไข hotel facilities ใน hotel

รันคำสั่งจาก:

1. *SQL/update/update_room_type_facilities.sql*
2. *SQL/sample_data/sample_update_room_facil.sql (ข้อมูลทดสอบ)*

คำสั่ง:

คำสั่ง แก้ไข hotel facilities จากไฟล์ SQL/update/update_room_type_facilities.sql

```
CREATE OR REPLACE PROCEDURE update_room_type_facilities(
    update_hotel_id INT,
    update_room_type VARCHAR(64),
    update_room_facilities VARCHAR(256)
)
LANGUAGE plpgsql
AS $$
BEGIN
    UPDATE room_type_facilities
        SET room_facilities = update_room_facilities
        WHERE hotel_id = update_hotel_id AND room_type = update_room_type;
    RAISE NOTICE 'Room Facilities updated successfully';
END; $$;
```

ก่อนแก้

	hotel_id integer	room_type character varying (64)	min_price integer	max_price integer	room_facilities character varying (256)
1	10000	Cheap	800	1500	Single bed, ceiling fan, shared bathroom, free WiFi, basic toiletries, no minibar
2	10000	Medium	1800	2500	Queen bed, air conditioning, private bathroom, TV, WiFi, minibar, toiletries
3	10000	Expensive	3000	4000	King bed, air conditioning, private jacuzzi, smart TV, high-speed WiFi, luxury toiletries, balcony, 24/7 room service
4	10001	Cheap	200	300	Twin bed, heater, shared bathroom, free WiFi, basic toiletries, no TV
5	10001	Medium	400	500	Queen bed, heating, private bathroom, TV, WiFi, minibar, toiletries
6	10001	Expensive	600	700	King bed, fireplace, private jacuzzi, smart TV, high-speed WiFi, luxury toiletries, scenic balcony, 24/7 room service

หลังแก้

	hotel_id integer	room_type character varying (64)	min_price integer	max_price integer	room_facilities character varying (256)
1	10000	Expensive	3000	4000	King bed, air conditioning, private jacuzzi, smart TV, high-speed WiFi, luxury toiletries, balcony, 24/7
2	10000	Medium	1800	2500	Queen bed, air conditioning, private bathroom, TV, WiFi, minibar, toiletries
3	10000	Cheap	800	1500	Basic amenities: Twin beds, air conditioning, shared bathroom, free Wi-Fi
4	10001	Expensive	600	700	Luxury amenities: King-size bed, private balcony, whirlpool tub, in-room dining, 24/7 butler service
5	10001	Cheap	200	300	Twin bed, heater, shared bathroom, free WiFi, basic toiletries, no TV
6	10001	Medium	400	500	Queen bed, heating, private bathroom, TV, WiFi, minibar, toiletries

5.) คำสั่งที่เกี่ยวข้องกับการ DELETE

5.1) ลบข้อมูล booking

คำอธิบาย: ฟังก์ชัน Delete_Booking คือฟังก์ชันที่ใช้ได้ทั้งกับ admin และ registered user เป็นฟังก์ชันที่รับ parameter 2 ตัว ได้แก่ delete_booking_id INT และ deleter_id INT สำหรับ admin จะลบได้ทุก booking_transaction แต่ registered user จะลบได้เฉพาะ booking_transaction ที่ตนเองเป็นผู้จองได้เท่านั้น โดย

- ฟังก์ชันจะตรวจสอบ deleter_id ที่รับเข้ามา มีสิทธิ์ admin หรือไม่
  ถ้ามีฟังก์ชันจะลบข้อมูลการจองโรงแรมที่มี booking_id ตรงกับ delete_booking_id ที่รับเข้ามา
- ตรวจสอบ deleter_id ที่รับเข้ามาแล้วไม่มีสิทธิ์ admin แต่เป็นผู้ใช้ที่เป็นเจ้าของ delete_id ที่รับเข้ามา ฟังก์ชันจะลบข้อมูลการจองโรงแรมที่มี booking_id ตรงกับ delete_booking_id และจะต้องมี user_id ของผู้จองตรงกับ deleter_id ที่รับเข้ามา
- ถ้าไม่เป็นไปตามเงื่อนไขก่อนหน้านี้ ฟังก์ชันจะไม่สามารถลบข้อมูลการจองได้

รันคำสั่งจาก:

1. *SQL/delete/delete_booking.sql*
2. *SQL/sample_data/sample_delete_booking.sql (ข้อมูลทดสอบ)*

คำสั่ง:

คำสั่ง ลบข้อมูลการจองจากไฟล์ SQL/delete/delete_booking.sql

```
CREATE OR REPLACE PROCEDURE delete_booking(
delete_booking_id INT,
deleter_id INT
)
LANGUAGE plpgsql
AS $$
BEGIN
    IF deleter_id >= 900000000 THEN

        DELETE FROM booking_log
            WHERE booking_id = delete_booking_id;

        DELETE FROM booking_transaction
            WHERE booking_id = delete_booking_id;

    ELSEIF deleter_id = (SELECT user_id FROM booking_transaction WHERE booking_id =
delete_booking_id ) THEN

        delete FROM booking_log
            WHERE booking_id = delete_booking_id;

        delete FROM booking_transaction
            WHERE booking_id = delete_booking_id;
```

```
    ELSE raise exception 'Deleter try to delete other user booking';
    END IF;
END;$$
```

ผลการรัน:

ก่อนลบ

	booking_id [PK] integer	booking_date date	duration integer	check_in_date date	check_out_date date	user_id integer	hotel_id integer	room_id integer
1	100000001	2025-02-10	2	2025-03-10	2025-03-12	900000001	10000	201
2	100000002	2025-02-10	2	2025-02-21	2025-02-23	900000001	10001	202
3	100000003	2025-02-10	1	2025-04-11	2025-04-12	900000001	10002	203
4	100000004	2025-02-10	2	2025-02-19	2025-02-21	900000001	10003	201
5	100000005	2025-02-10	2	2025-06-10	2025-06-12	900000001	10004	202
6	100000006	2025-02-10	2	2025-02-10	2025-02-12	900000001	10005	203

หลังลบ

	booking_id [PK] integer	booking_date date	duration integer	check_in_date date	check_out_date date	user_id integer	hotel_id integer	room_id integer
1	100000002	2025-02-10	2	2025-02-21	2025-02-23	900000001	10001	202
2	100000004	2025-02-10	2	2025-02-19	2025-02-21	900000001	10003	201
3	100000006	2025-02-10	2	2025-02-10	2025-02-12	900000001	10005	203
4	100000008	2025-02-10	2	2025-02-10	2025-02-12	900000001	10007	202
5	100000010	2025-02-10	2	2025-02-10	2025-02-12	900000001	10009	201

คำอธิบาย: ฟังก์ชัน delete__hotel คือฟังก์ชันที่ใช้ได้เฉพาะ admin เท่านั้น โดยเป็นฟังก์ชันที่รับ parameter 2 ตัว ได้แก่ admin_id INT และ delete_hotel_id INT โดย

- ฟังก์ชันจะตรวจสอบว่า admin_id ที่รับเข้ามา มีสิทธิ์ admin หรือไม่
  ถ้าไม่มีจะไม่สามารถลบข้อมูลของโรงแรมได้
- หลังจากการตรวจสอบสิทธิ์ admin ฟังก์ชันจะตรวจสอบว่า delete_hotel_id มีอยู่จริงหรือไม่
  ถ้าไม่มีจะไม่สามารถลบข้อมูลได้
- ถ้า admin_id และ delete_hotel_id ถูกต้องตามเงื่อนไขแล้ว ฟังก์ชันจะลบข้อมูลของโรงแรมที่มี hotel_id ตรงกับ parameter ที่รับมา

รันคำสั่งจาก:

1. *SQL/delete/delete_hotel.sql*
2. *SQL/sample_data/sample_delete_hotel.sql (ข้อมูลทดสอบ)*

คำสั่ง:

คำสั่งลบข้อมูล hotel จากไฟล์ SQL/delete/delete_hotel.sql

```
CREATE OR REPLACE PROCEDURE delete_hotel(
    admin_id INT,
    delete_hotel_id INT
)
LANGUAGE plpgsql
AS $$
DECLARE
    is_admin INT;
    exists_hotel INT;
BEGIN
    SELECT COUNT(*) INTO is_admin FROM admin WHERE user_id = admin_id;
    IF is_admin = 0 THEN
        RAISE EXCEPTION 'Permission denied: Only admins can delete
hotels';
    END IF;

    SELECT COUNT(*) INTO exists_hotel FROM hotel WHERE hotel_id =
delete_hotel_id;
    IF exists_hotel = 0 THEN
        RAISE EXCEPTION 'Error: Hotel does not exist';
    END IF;

    DELETE FROM hotel_log WHERE hotel_id = delete_hotel_id;
    DELETE FROM booking_log WHERE booking_id IN (SELECT booking_id FROM
booking_transaction WHERE hotel_id = delete_hotel_id);
    DELETE FROM booking_transaction WHERE hotel_id = delete_hotel_id;
    DELETE FROM room_type_facilities WHERE hotel_id = delete_hotel_id;
    DELETE FROM room WHERE hotel_id = delete_hotel_id;
    DELETE FROM hotel WHERE hotel_id = delete_hotel_id;

    RAISE NOTICE 'Hotel deleted successfully';
END; $$
```

ผลการรัน:

ก่อนลบ

	hotel_id [PK] integer	map_url character varying	hotel_name character varying (256)	hotel_phonenumber character varying (10)	check_in_time time without time zone	check_out_time time without time zone	hotel_facilities character varying (512)
1	10000	http://example.com/max1	Super Skyscraper Hotel	0111111111	12:00:00	10:00:00	WiFi, Pool, Valet Parking, Rooftop Bar, Business Center, Gym, Spa, Indoor Pool
2	10001	http://example.com/max2	Luxury Haven Resort	0222222222	13:30:00	11:00:00	WiFi, Pool, Valet Parking, Butler Service, Private Beach, Fine Dining, Spa, Private Caban
3	10002	[null]	Beachfront Resort	0888888888	15:30:00	11:00:00	WiFi, Beach Access, Restaurant, Spa
4	10003	http://example.com/hotel4	Mountain View Lodge	0999999999	13:00:00	10:30:00	WiFi, Hiking Trails, Free Breakfast
5	10004	http://example.com/latecheckin	Beachfront Resort	0888888888	15:30:00	11:00:00	WiFi, Beach Access, Restaurant, Spa
6	10005	http://example.com/hotel_standar...	Sunset Paradise Resort	0555555555	15:00:00	11:30:00	WiFi, Pool, Beachfront, Spa, Restaurant, Bar, Kids Club
7	10006	http://example.com/max2	Luxury Haven Resort	0222222222	13:30:00	11:00:00	WiFi, Pool, Valet Parking, Butler Service, Private Beach, Fine Dining, Spa, Private Caban
8	10007	http://example.com/max3	Ultimate Tower Hotel	0333333333	15:00:00	12:30:00	WiFi, Pool, Valet Parking, Rooftop Bar, Nightclub, Movie Theater, Golf Course, Tennis Cc
9	10008	http://example.com/max4	Midnight Express Hotel	0444444444	23:59:00	00:01:00	WiFi, Pool, Valet Parking, 24-hour Room Service, Casino, Spa, Gym, Live Entertainment
10	10009	http://example.com/hotel_standar...	Sunset Paradise Resort	0555555555	15:00:00	11:30:00	WiFi, Pool, Beachfront, Spa, Restaurant, Bar, Kids Club

หลังลบ

	hotel_id [PK] integer	map_url character varying	hotel_name character varying (256)	hotel_phonenumber character varying (10)	check_in_time time without time zone	check_out_time time without time zone	hotel_facilities character varying (512)
1	10000	http://example.com/max1	Super Skyscraper Hotel	0111111111	12:00:00	10:00:00	WiFi, Pool, Valet Parking, Rooftop Bar, Business Center, Gym, Spa, Indoor Pool
2	10002	[null]	Beachfront Resort	0888888888	15:30:00	11:00:00	WiFi, Beach Access, Restaurant, Spa
3	10004	http://example.com/latecheck...	Beachfront Resort	0888888888	15:30:00	11:00:00	WiFi, Beach Access, Restaurant, Spa
4	10006	http://example.com/max2	Luxury Haven Resort	0222222222	13:30:00	11:00:00	WiFi, Pool, Valet Parking, Butler Service, Private Beach, Fine Dining, Spa, Private Caban
5	10008	http://example.com/max4	Midnight Express Hotel	0444444444	23:59:00	00:01:00	WiFi, Pool, Valet Parking, 24-hour Room Service, Casino, Spa, Gym, Live Entertainment

SQL Complex Query

ยกตัวอย่างการ query เพื่อแสดงยอดการจองโรงแรมที่เยอะที่สุดมา10อันดับ (hotel_id , hotel_name)

```
SELECT  h.hotel_id,
        h.hotel_name,
        COUNT(DISTINCT bt.booking_id) AS total_bookings,
        ROUND(AVG(bt.duration)) AS avg_stay_duration
FROM hotel h
LEFT JOIN booking_transaction bt ON h.hotel_id = bt.hotel_id
GROUP BY h.hotel_id, h.hotel_location
ORDER BY total_bookings DESC
LIMIT 10;
```

	hotel_id [PK] integer	hotel_name character varying (256)	total_bookings bigint	avg_stay_duration numeric
1	10009	Sunset Paradise Resort	3	2
2	10001	Luxury Haven Resort	2	2
3	10002	Beachfront Resort	2	2
4	10003	Mountain View Lodge	2	2
5	10004	Beachfront Resort	2	2
6	10000	Super Skyscraper Hotel	2	2
7	10006	Luxury Haven Resort	2	2
8	10007	Ultimate Tower Hotel	2	2
9	10008	Midnight Express Hotel	2	2
10	10005	Sunset Paradise Resort	2	2

Document based design schema

```
{
  "title": "booking_transaction",
  "required": ["_id", "booking_id", "booking_date", "duration", "check_in_date", "check_out_date",
"hotel_id", "room_id", "user"],
  "properties": {
      "_id": { "bsonType": "objectId" },
      "booking_id": { "bsonType": "string" },
      "booking_date": { "bsonType": "date" },
      "duration": { "bsonType": "int" },
      "check_in_date": { "bsonType": "date" },
      "check_out_date": { "bsonType": "date" },
      "hotel_id": { "bsonType": "string" },
       "room_id" : { "bsonType": "string"}
      "user": {
        "bsonType": "object",
        "properties": {
          "user_id": { "bsonType": "string" },
          "user_type": { "bsonType": "string", "enum": ["admin", "registered", "unregistered"] }
        }
    }
  }
}
```