CSI2132 Database I

eHotel-Course Project:

Entity Relationship Model Related Design

Deliverable 1 Report

Roy, Rui 300176548

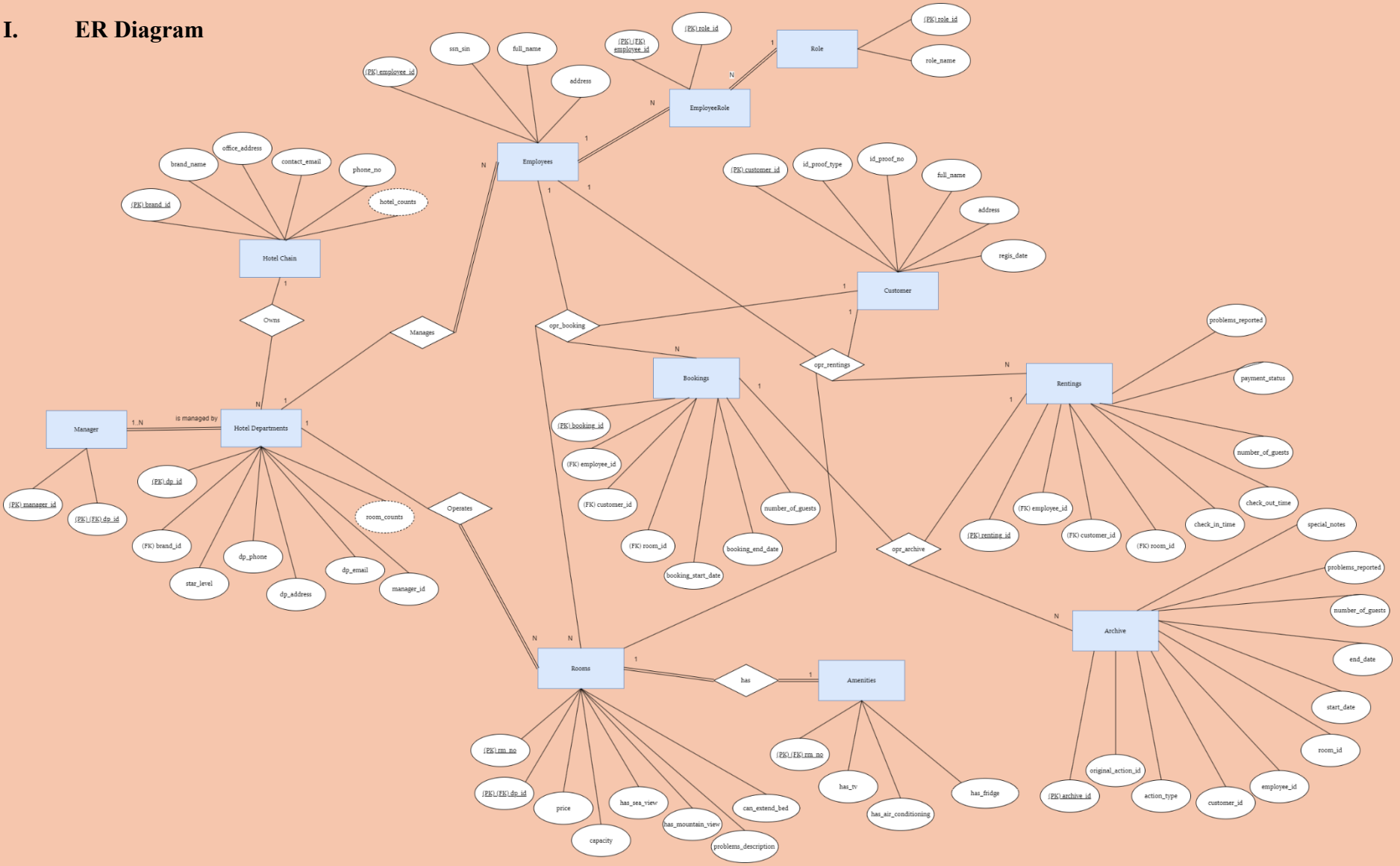
Yucheng, Chen 300194614

University of Ottawa

Faculty of Engineering

February 16, 2024

I. ER Diagram


Role
{PK} role_id
role_name
EmployeeRole
{PK} role_id
{PK}{FK} employee_id
Employees
{PK} employee_id
ssn_sin
full_name
address
Hotel Chain
{PK} brand_id
brand_name
office_address
contact_email
phone_no
hotel_counts
Owns
Manages
Manager
{PK} manager_id
{PK}{FK} dp_id
is managed by
Hotel Departments
{PK} dp_id
{FK} brand_id
star_level
dp_phone
dp_address
dp_email
manager_id
room_counts
Operates
Customer
{PK} customer_id
id_proof_type
id_proof_no
full_name
address
regis_date
opr_booking
opr_rentings
Bookings
{PK} booking_id
(FK) employee_id
(FK) customer_id
(FK) room_id
booking_start_date
booking_end_date
number_of_guests
Rentings
{PK} renting_id
(FK) employee_id
(FK) customer_id
(FK) room_id
problems_reported
payment_status
number_of_guests
check_out_time
check_in_time
special_notes
opr_archive
Archive
{PK} archive_id
problems_reported
number_of_guests
end_date
start_date
room_id
original_action_id
action_type
customer_id
employee_id
Rooms
{PK} rm_no
{PK}{FK} dp_id
price
capacity
has_sea_view
has_mountain_view
problems_description
can_extend_bed
has
Amenities
{PK}{FK} rm_no
has_tv
has_air_conditioning
has_fridge

Justification for the ER Diagram:

The ER diagram is designed to comprehensively represent the data structure of a hotel management system. Key entities such as *Hotel Chain, Hotel, Room, Booking, Renting, Customer, Employee,* and *Amenities* are included to reflect all aspects of hotel operations.

Entities: Each entity represents a distinct aspect of the hotel's operations, ensuring that data is organized logically and efficiently. Attributes are selected to capture essential details required for the system's functionality.

For some special condition, like “Every hotel needs to have a manager,” Managers entity is specifically designed to satisfy the business requirement that every hotel must have a designated manager. The Managers entity enforces data integrity by using a foreign key that references the primary key of the Employees entity. This ensures that every manager is an existing employee.

Relationships: Defined relationships and cardinalities between entities reflect real-world interactions and business rules, such as customers making bookings and employees managing rentals.

Archiving: The inclusion of an Archive entity allows for historical data preservation, aligning with the business need to maintain records even after associated entities are deleted.

II. Relational Database Schema

```
-- Hotel Chain
CREATE TABLE HotelChain (
    brand_id INT PRIMARY KEY,
    brand_name VARCHAR(255),
    office_address VARCHAR(255),
    contact_email VARCHAR(255),
    phone_no VARCHAR(20)
    hotel_counts INT
);

-- Hotel
CREATE TABLE HotelDepartments (
    dp_id INT PRIMARY KEY,
    brand_id INT,
    star_level INT,
    dp_phone VARCHAR(20),
    dp_address VARCHAR(255),
    dp_email VARCHAR(255),
    manager_id INT,
    room_counts INT,
    FOREIGN KEY (brand_id) REFERENCES HotelChain(brand_id),
    FOREIGN KEY (manager_id) REFERENCES Manager(employee_id)
);

-- Employees
CREATE TABLE Employees (
    employee_id INT PRIMARY KEY,
    ssn_sin VARCHAR(20),
    full_name VARCHAR(255),
    address VARCHAR(255),
    FOREIGN KEY (role_id) REFERENCES Role(role_id)
);

-- Role
CREATE TABLE Role (
    role_id INT PRIMARY KEY,
    role_name VARCHAR(255)
);
```

```
-- EmployeeRole (Junction Table for many-to-many relationship
between Employees and Roles)
CREATE TABLE EmployeeRole (
    employee_id INT,
    role_id INT,
    PRIMARY KEY (employee_id),
    FOREIGN KEY (employee_id) REFERENCES Employees(employee_id),
    FOREIGN KEY (role_id) REFERENCES Role(role_id)
);

-- Customer
CREATE TABLE Customer (
    customer_id INT PRIMARY KEY,
    full_name VARCHAR(255),
    address VARCHAR(255),
    id_proof_type VARCHAR(50),
    id_proof_no VARCHAR(50),
    regis_date DATE
);

-- Rooms
CREATE TABLE Rooms (
    room_no INT PRIMARY KEY,
    dp_id INT,
    price DECIMAL(10, 2),
    capacity INT,
    has_sea_view BOOLEAN,
    has_mountain_view BOOLEAN,
    can_extend_bed BOOLEAN,
    problems_description TEXT,
    FOREIGN KEY (dp_id) REFERENCES Hotel(dp_id)
);
-- Rooms
CREATE TABLE Amenities (
    rm_no INT PRIMARY KEY,
    has_tv BOOLEAN,
    has_air_conditioning BOOLEAN,
    has_fridge BOOLEAN,
    FOREIGN KEY (rm_no) REFERENCES Rooms(rm_no)
);

-- Bookings
CREATE TABLE Booking (
    booking_id INT PRIMARY KEY,
```

```
    customer_id INT,
    room_id INT,
    employee_id INT,
    number_of_guests INT,
    booking_start_date DATE,
    booking_end_date DATE,
    FOREIGN KEY (customer_id) REFERENCES Customer(customer_id),
    FOREIGN KEY (room_id) REFERENCES Rooms(rm_id),
    FOREIGN KEY (employee_id) REFERENCES Employees(employee_id)
);

-- Rentings
CREATE TABLE Renting (
    renting_id INT PRIMARY KEY,
    customer_id INT,
    room_id INT,
    employee_id INT,
    check_in_date DATE,
    check_out_date DATE,
    number_of_guests INT,
    payment_status VARCHAR(100),
    FOREIGN KEY (customer_id) REFERENCES Customer(customer_id),
    FOREIGN KEY (room_id) REFERENCES Rooms(room_id),
    FOREIGN KEY (employee_id) REFERENCES Employees(employee_id)
);

-- Archive (Could be for either Bookings or Rentings)
CREATE TABLE Archive (
    archive_id INT PRIMARY KEY,
    original_action_id INT,
    action_type VARCHAR(50), -- 'Booking' or 'Renting'
    customer_id INT,
    room_id INT,
    employee_id INT,
    start_date DATE,
    end_date DATE,
    number_of_guests INT,
    problem_reported VARCHAR(255),
    special_notes VARCHAR(255),
    archive_date DATE
    -- No foreign keys to allow for historical preservation even if
original records are deleted
);
```

III. Integrity Constraints

1. Hotel Chain Constraints:

hotel_chain_id (PK): Ensures uniqueness for each hotel chain.

office_address, *contact_email*, *phone_no*: Not Null constraints to ensure every hotel chain has contact information.

hotel_counts: A Check constraint to ensure the value is a non-negative integer.

2. Hotel Departments Constraints:

dp_id (PK): Uniquely identifies each hotel.

brand_id (FK): Ensures each hotel is associated with a hotel chain.

star_level: A Check constraint to ensure it falls within the allowed range (e.g., 1 to 5).

room_counts, *address*, *hotel_email*, *hotel_phone*: Not Null constraints for essential hotel details.

3. Room Constraints:

rm_id (PK): Uniquely identifies each room within a hotel.

hotel_id (PK) (FK): Ensures each room is associated with a hotel.

price: A Check constraint to ensure the price is a positive value.

capacity: Not Null to guarantee every room's capacity is recorded.

has_sea_view, *has_mountain_view*, *can_extend_bed*: Boolean fields, defaulting to false.

problems_description: Allows Null to handle rooms without problems.

4. Customer Constraints:

customer_id (PK): Uniquely identifies each customer.

full_name, *address*, *id_type*, id_number, regis_date: Not Null constraints to ensure all required customer information is captured.

5. **Employee Constraints:**

employee_id (PK): Uniquely identifies each employee.

full_name, *address*, *ssn_sin*: Not Null constraints to ensure all required employee information is captured.

role: Not Null to guarantee that each employee's role is defined.

6. **Manager Constraints:**

Each hotel must have one and only one manager. This can be enforced by a Unique constraint on a manager_id field in the Hotel table.

manager_id (FK): References an employee_id from the Employees table.

7. **Booking and Renting Constraints:**

booking_id, *renting_id* (PK): Ensure uniqueness for bookings and rentings.

customer_id, *room_id*, *employee_id* (FK): Link bookings and rentings to customers, rooms, and employees.

start_date, *end_date*: Not Null to capture the duration of bookings and rentings.

Archiving process should ensure foreign key references are not violated if related records are deleted.