Entity Relationship Diagram (ERD) & Normalization Documentation

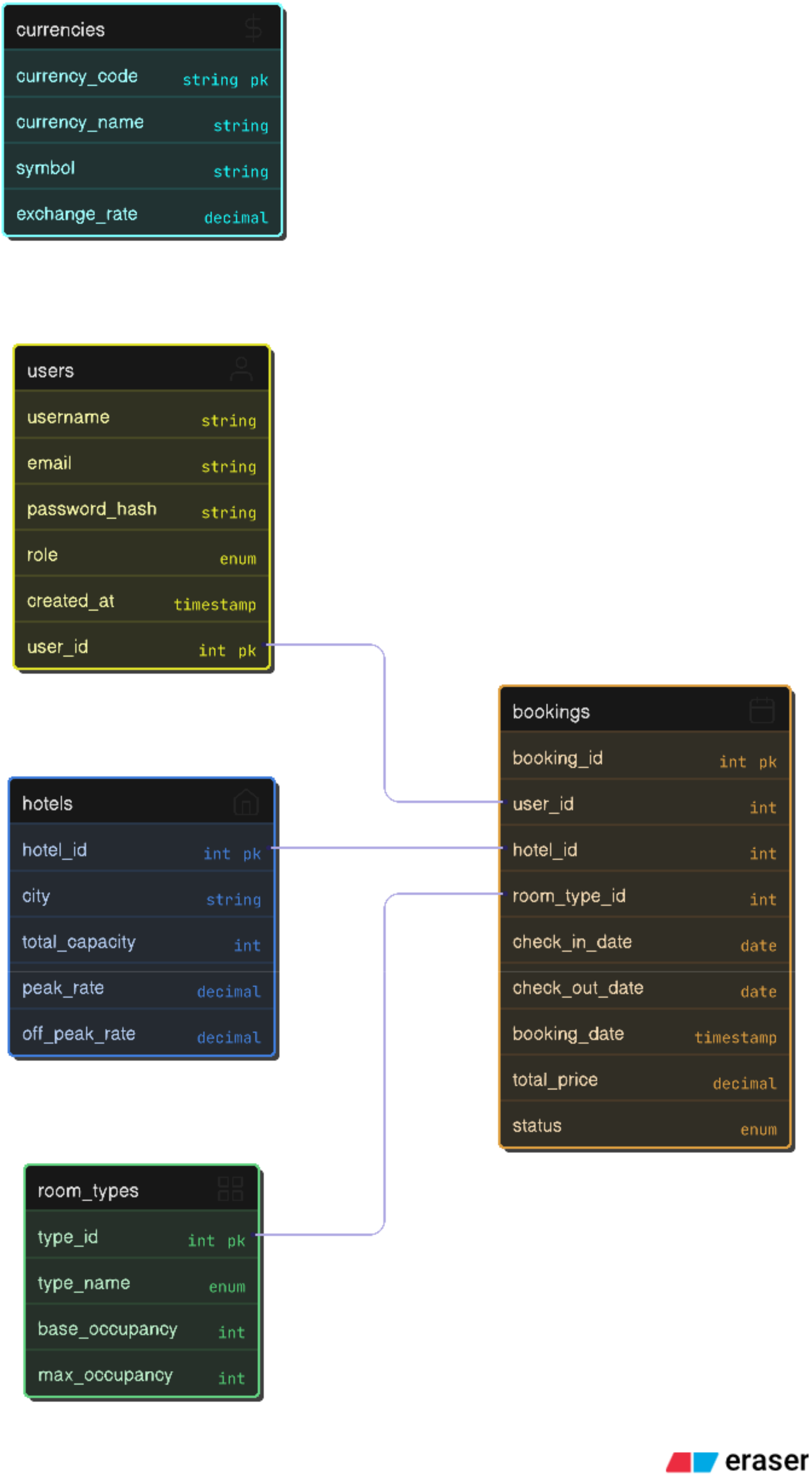
Student ID: 24071105
Student Name: Reya Adhikari

1. Entity Relationship Diagram (ERD)

The following diagram illustrates the database structure and relationships between entities in the World Hotels (WH) Booking System.

Relationships Explained

- **Users to Bookings (1:N)**: A single user can make multiple bookings, but each booking belongs to exactly one user.
- **Hotels to Bookings (1:N)**: A hotel can have multiple bookings associated with it (up to its capacity), but a single booking record refers to a stay at one specific hotel.
- **Room_Types to Bookings (1:N)**: A room type (e.g., “Standard”) is referenced by many bookings. This lookup table approach avoids repeating room details (occupancy limits) in every booking record.
- **Currencies**: This is a standalone lookup table used for the Pricing Engine / Display conversion features. It does not have a foreign key relationship with bookings as the base accounting is done in GBP (£), and conversion happens at the application level for display.



currencies	
currency_code	string pk
currency_name	string
symbol	string
exchange_rate	decimal

users	
username	string
email	string
password_hash	string
role	enum
created_at	timestamp
user_id	int pk

hotels	
hotel_id	int pk
city	string
total_capacity	int
peak_rate	decimal
off_peak_rate	decimal

room_types	
type_id	int pk
type_name	enum
base_occupancy	int
max_occupancy	int

bookings	
booking_id	int pk
user_id	int
hotel_id	int
room_type_id	int
check_in_date	date
check_out_date	date
booking_date	timestamp
total_price	decimal
status	enum

Entity Relationship Diagram

2. Normalization Process (3NF Compliance)

The database schema has been designed to strictly adhere to the Third Normal Form (3NF) to minimize redundancy and prevent data anomalies.

First Normal Form (1NF)

Requirement: Atomic values, unique columns, primary keys, no repeating groups.

- **Implementation**:
  - All tables (`users`, `hotels`, `bookings`, etc.) have a Primary Key (`id`).
  - No columns contain lists of data (e.g., we do not store multiple phone numbers in a "phones" column).
  - No repeating groups (e.g., we do not have columns like `Day1_Price`, `Day2_Price` in the bookings table).

Second Normal Form (2NF)

Requirement: Must be in 1NF, and all non-key attributes must be fully functionally dependent on the Primary Key (no partial dependencies).

- **Implementation**:
  - In the `bookings` table, attributes like `check_in_date` and `total_price` depend entirely on the `booking_id`. They do not depend on just part of a composite key (since we use a single surrogate Primary Key `booking_id`).
  - Hotel details (like `city` and `peak_rate`) are stored in the `hotels` table, dependent on `hotel_id`. If they were stored in `bookings`, they would depend on `hotel_id` (which is only part of the booking context), violating 2NF logic if we used a composite key of (User+Hotel). By splitting them, we ensure full dependency on their respective table PKs.

Third Normal Form (3NF)

Requirement: Must be in 2NF, and have no transitive dependencies (non-key attributes strictly depend on the Primary Key, not on other non-key attributes).

- **Implementation**:
  - **Room Types**: We extracted room configuration into a separate `room_types` table.
    - *Violation Scenario*: If we stored `max_occupancy` in the `bookings` table, it would depend on the `room_type` (e.g., "Family" rooms always hold 4). This would be a transitive dependency: `booking_id` -> `room_type` -> `max_occupancy`.
    - *Resolution*: We store `max_occupancy` in `room_types` (dependent on `type_id`) and just reference `room_type_id` in `bookings`.
  - **Pricing**: We do not store "Hotels' base rates" in the `bookings` table. We store the `total_price` (which is transaction-specific history) in `bookings`, but the master rate data (`peak_rate`, `off_peak_rate`) lives in `hotels`, dependent vertically on `hotel_id`.