MET CS 669 Database Design and Implementation for Business

Project Name: ABC's Hotel Management System
Submitted By: Ratna Meena Shivakumar

Current Iteration: Iteration 4
Full DBMS Physical ERD – You define the attributes for your database design and add them to your DBMS physical ERD.
Normalization – You normalize your DBMS physical ERD to reduce or eliminate data redundancy.
Tables and Constraints – You create your tables and constraints in SQL.
Index Placement and Creation – To speed up performance, you identify columns needing indexes for your database, then create them in SQL.

This database aims to develop a hotel management system for **ABC Hotels** to manage the work efficiently. This project's primary focus is to enable online booking for customers and allows the managers to track the booking details and employee records. This database will contain all the records of the employees, customers, rooms, and other services offered. The customer can create an account, login and book the desired rooms and other facilities involved or make modifications if applicable to the booking made. The manager can easily track what kind of room the customer has booked, enabling fast and easy retrieval of guest records and data for fast reference activities.

The Scenarios in which this database will be used are as follows

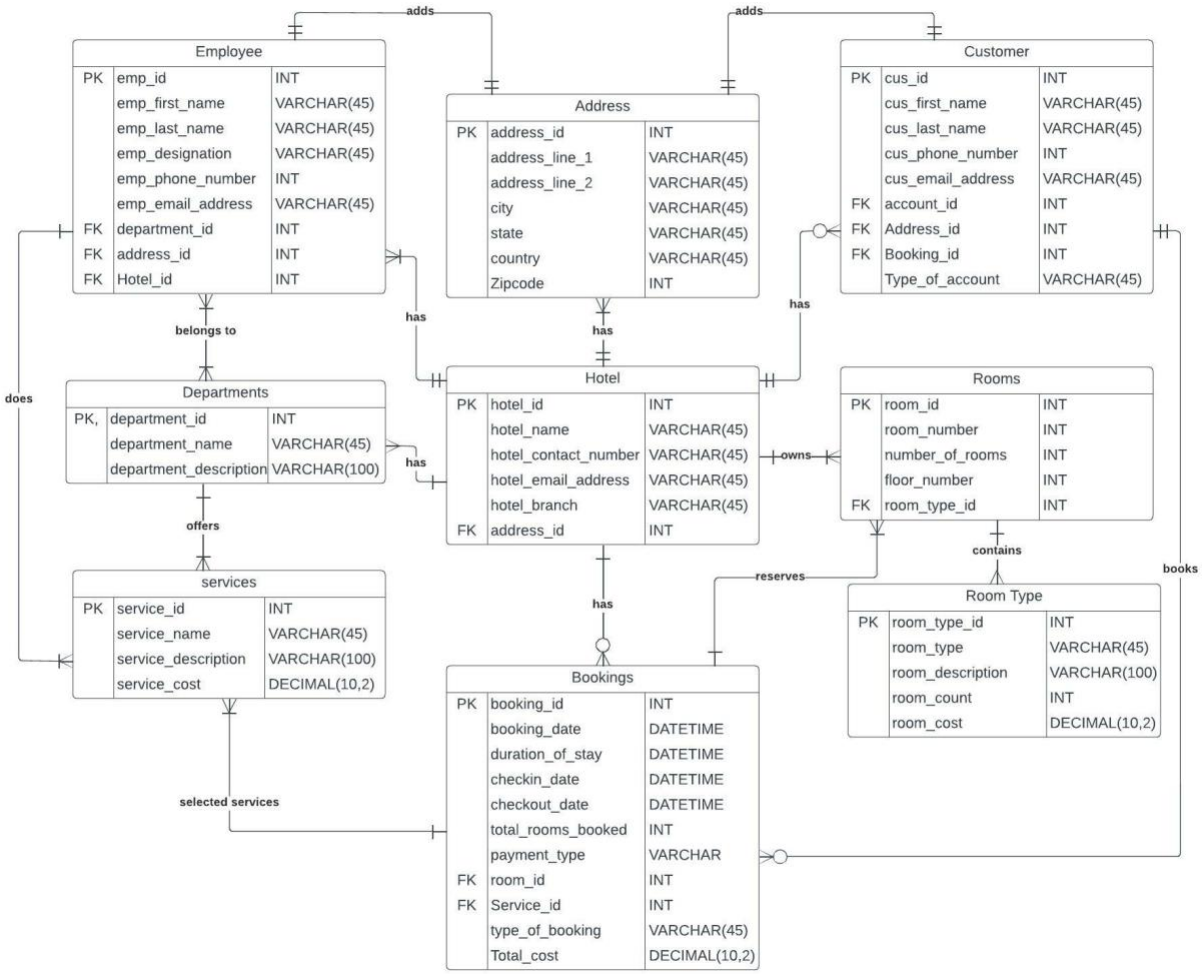
(i) A Customer wants to book a hotel room; he can create an account to avail offers and book the room.
(ii) A Customer can check for different type of hotel rooms and the type of services offered by the hotel
(iii) Employee working at the hotel can login and update his personal records.
(iv) Employee can check for the shift timings and the record the timing he worked for by logging in.
(v) Employee can check for the assigned service as requested by the customer
(vi) Employee with the role as manager can assign work to other employee.

Attributes:

Table	Attribute	Data type	Reasoning
Hotel	hotel_name	VARCHAR(45)	This will be helpful to identify the hotel name
Hotel	hotel_contact_number	INT	This will be helpful for the communication purposes when a hotel cannot be reached through mails.
Hotel	hotel_email_addresss	VARCHAR(45)	This is necessary while sending email to the hotel.
Hotel	hotel_branch	VARCHAR(45)	This will be helpful to identify which branch the hotel belongs to.
Employee	emp_first_name	VARCHAR(45)	This is necessary for displaying the person's name on screens and addressing them when sending them emails or other communications
Employee	emp_last_name	VARCHAR(45)	This is necessary for displaying the person's name on screens and addressing them when sending them emails or other communications
Employee	emp_designation	VARCHAR(45)	This denotes the role of the employee.
Employee	emp_phone_number	INT	This will be helpful for the communication purposes when a employee cannot be reached through mails.
Employee	emp_email_address	VARCHAR(45)	This is necessary while sending email to the employee
Customer	cus_first_name	VARCHAR(45)	This is necessary for displaying the person's name on screens and addressing them when sending them emails or other communications.
Customer	cus_last_name	VARCHAR(45)	This is necessary for displaying the person's name on screens and addressing them when sending them emails or other communications
Customer	cus_email_address	VARCHAR(45)	This is necessary while sending email to the customer
Customer	Cus_phone_number	INT	This will be helpful for the communication purposes when a customer cannot be reached through mails.

Customer	Type_of_account	VARCHAR(45)	Helps in identifying whether the account is basic or premium.
Address	address_line_1	VARCHAR(45)	First line of address denoting the house or apartment number.
Address	address_line_2	VARCHAR(45)	Second line of address denotes the area name.
Address	city	VARCHAR(45)	Specifies the city.
Address	state	VARCHAR(45)	Specifies the state.
Address	Country	VARCHAR(45)	Specifies the country.
Address	zipcode	INT	Specifies the zip code.
Departments	department_name	VARCHAR(45)	This will be helpful in segregating the departments
Departments	department_description	VARCHAR(45)	Describes the department.
services	service_name	VARCHAR(45)	Specifies the type of service.
services	service_description	VARCHAR(45)	Describes the type of service.
services	service_cost	DECIMAL(10,2)	Denotes the cost of the service.
Rooms	room_number	INT	It is necessary when uniquely identifying the room.
Rooms	number_of_rooms	INT	It is necessary when specifying the total number of rooms available in a specific type.
Rooms	floor_number	INT	Helps in identifying the floor number of the room.
Room Type	room_type	VARCHAR(45)	Helps in identifying the type of room.
Room Type	room_description	VARCHAR(45)	Describing the type of room.
Room Type	room_cost	DECIMAL(10,2)	Total cost of the room
Room Type	Room_count	INT	Denotes the number of rooms available in specific type.
Bookings	booking_date	DATE	Denotes the booking date.
Bookings	duration_of_stay	INT	Specifies the duration of stay,
Bookings	checkin_date	DATE	Specifies the checkin date,
Bookings	checkout_date	DATE	Specifies the checkout date
Bookings	total_rooms_booked	INT	Denotes the total number of rooms booked.
Bookings	payment_type	VARCHAR(45)	Specifies the payment type.
Bookings	type_of_booking	VARCHAR(45)	Identifies the type of booking is it is online or inperson.
Bookings	Total_cost	DECIMAL(10,2)	Specifies the total cost of the booking.

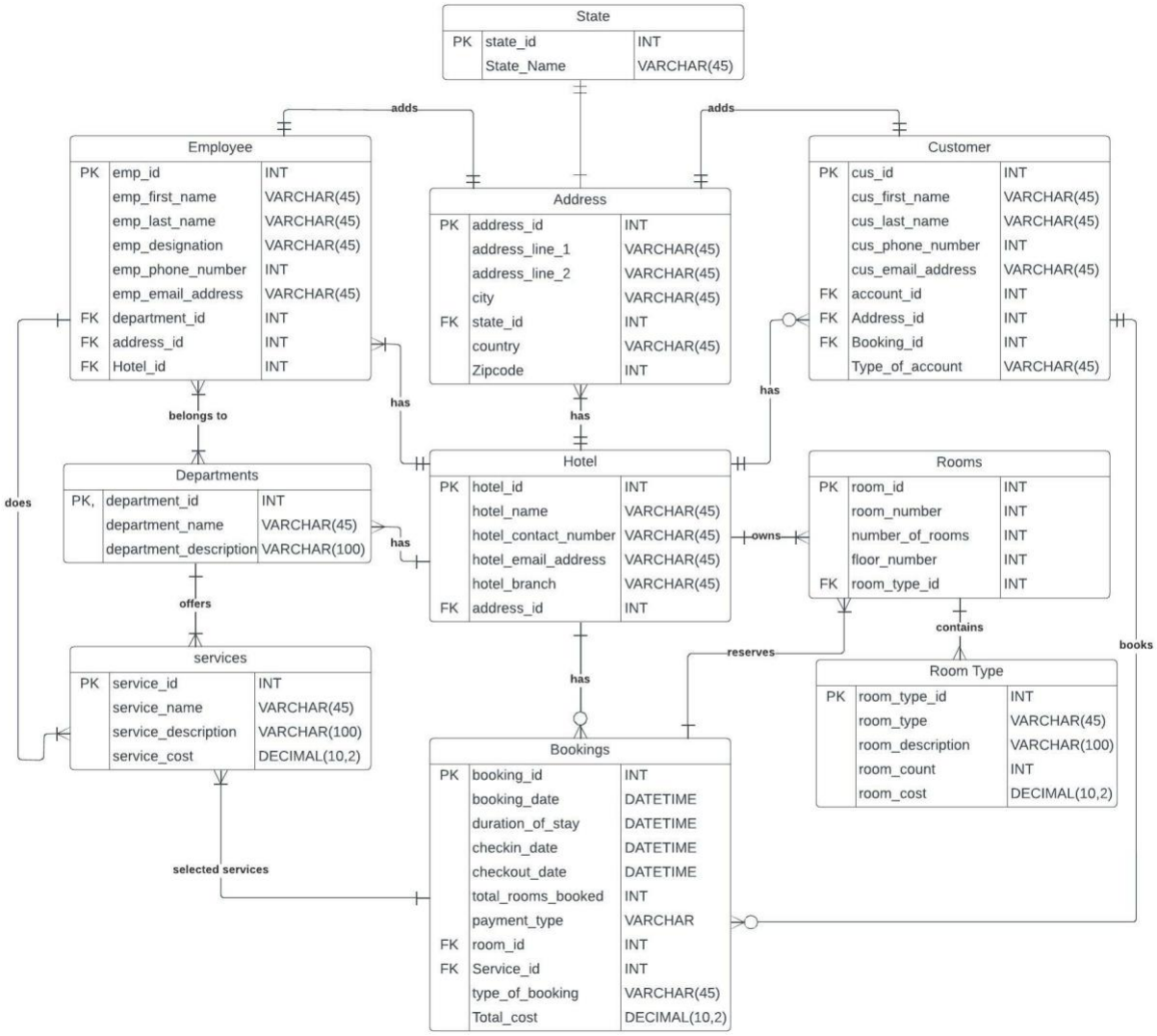
Full DBMS Physical ERD:



Normalization:

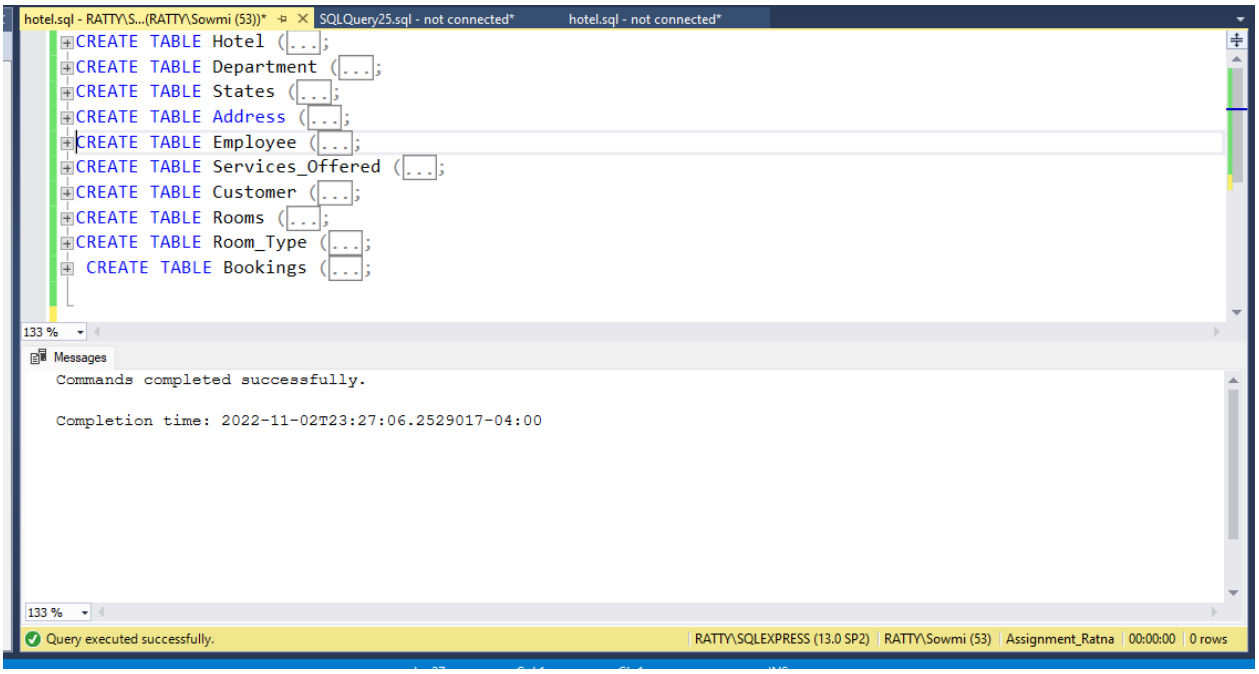
1) The table is normalized to 1NF since each table cell contains a single value and is unique.
2) The table is in 2NF since it satisfies the condition to be in 1NF such that Single Column Primary Key that is not functionally dependent on any subset of candidate key relation. Example: The primary key is not dependent on any other key that is belonging to the other table and there is only one primary key in each table. We will only be able to insert values into your foreign key that exist in the unique key in the parent table. This helps in referential integrity.
3) The table needs 3NF since the City and zip code is of transitive dependency. Each address has a state; each state may be associated with many addresses. The BCNF is done below in the table.

BCNF Normalized Physical ERD:



Tables and Constraints:

Creation of Tables.



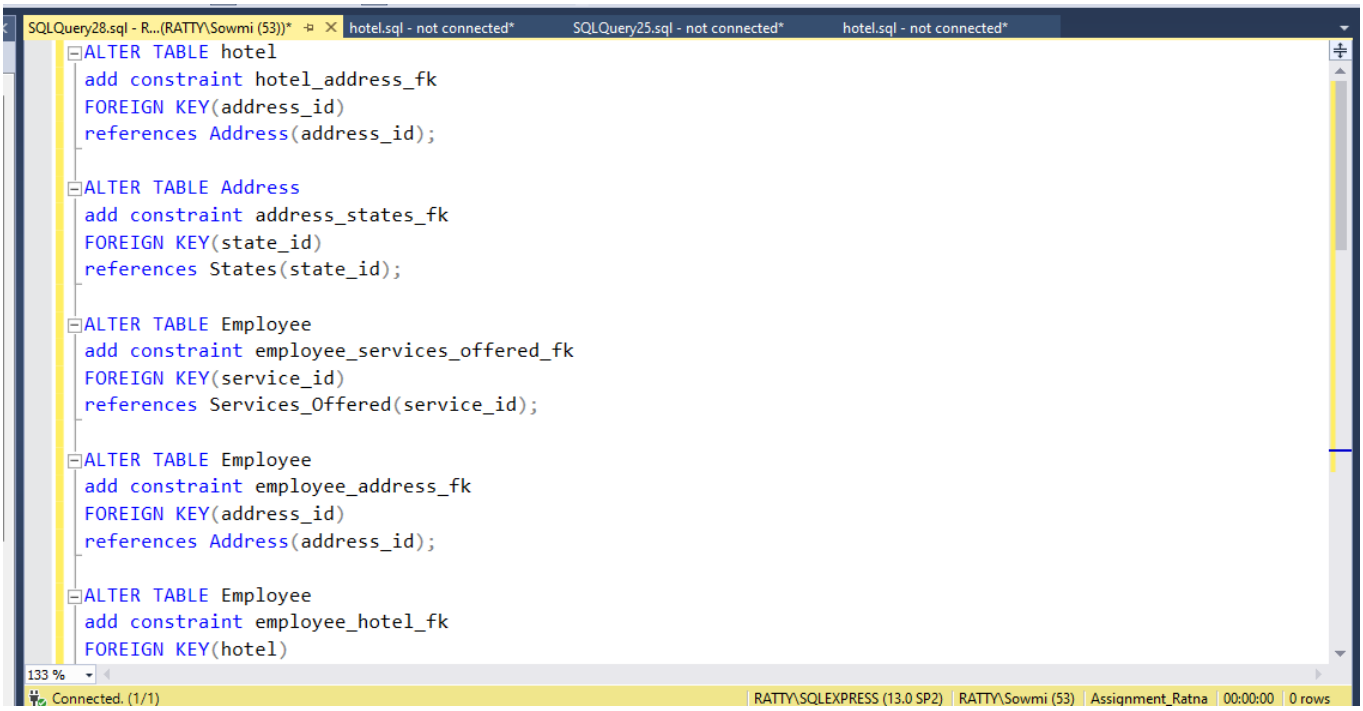
Constraint: Primary Key

PRIMARY KEY	DATA TYPE	DESCRIPTION
hotel_id	DECIMAL(12)	This is the primary key for the Table table
address_id	DECIMAL(12)	This is the primary key for the Address table
state_id	DECIMAL(12)	This is the primary key for the State table
emp_id	DECIMAL(12)	This is the primary key for the Employee table
cus_id	DECIMAL(12)	This is the primary key for the Customer table
room_id	DECIMAL(12)	This is the primary key for the Room table
room_type_id	DECIMAL(12)	This is the primary key for the Room type table
booking_id	DECIMAL(12)	This is the primary key for the Booking table
department_id	DECIMAL(12)	This is the primary key for the Department table
service_id	DECIMAL(12)	This is the primary key for the Service table

Constraint: Foreign Key

TABLE	FOREIGN KEY	DATA TYPE	DESCRIPTION
Address	address_states_fk	DECIMAL(12)	This foreign key in the address table references the States table. The Index is non-unique since many states can be on the same table.
Bookings	bookings_rooms_fk	DECIMAL(12)	This foreign key in the Bookings table references the rooms table. The Index is non-unique since many rooms can be booked on the same booking.
Bookings	bookings_services_offered_fk	DECIMAL(12)	This foreign key in the bookings table references the Services offered table. The Index is non-unique since many services can be under the same booking.
Customer	customer_bookings_fk	DECIMAL(12)	This foreign key in the Customer table references the bookings table. The Index is non-unique since many bookings can be under the same booking.
Customer	customers_address_fk	DECIMAL(12)	This foreign key in the Customer table references the address table. The Index is non-unique to refer the address of the customer.
Employee	Employee_address_fk	DECIMAL(12)	This foreign key in the Employee table references the address table. The Index is non-unique to refer the address of the customer.
Employee	Employee_hotel_fk	DECIMAL(12)	This foreign key in the employee table references the hotel table. The Index is non-unique to refer which hotel the employee is working for.
Employee	Employee_services_offered_fk	DECIMAL(12)	This foreign key in the employee table references the services_offered table. The Index is non-unique since many services can be handled by the same employee.
Hotel	Hotel_address_fk	DECIMAL(12)	This foreign key in the hotel table references the address table. The Index is non-unique since many address can be there for a hotel.
Rooms	Rooms_room_type_fk	DECIMAL(12)	This foreign key in the rooms table references the room_type table. The Index

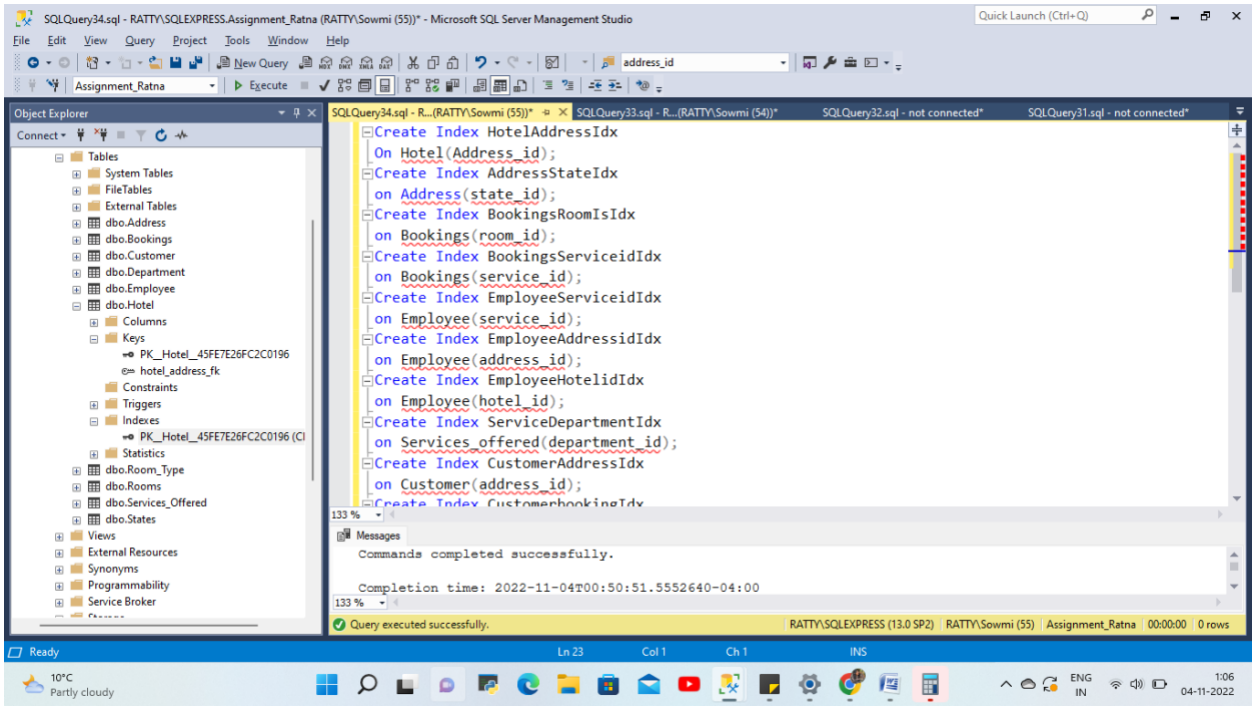
			is non-unique since many room can be of the same type.
Services_offered	Services_offered_departments_fk	DECIMAL(12)	This foreign key in the services_offered table references the department table. The Index is non-unique since many services can be in the same department.



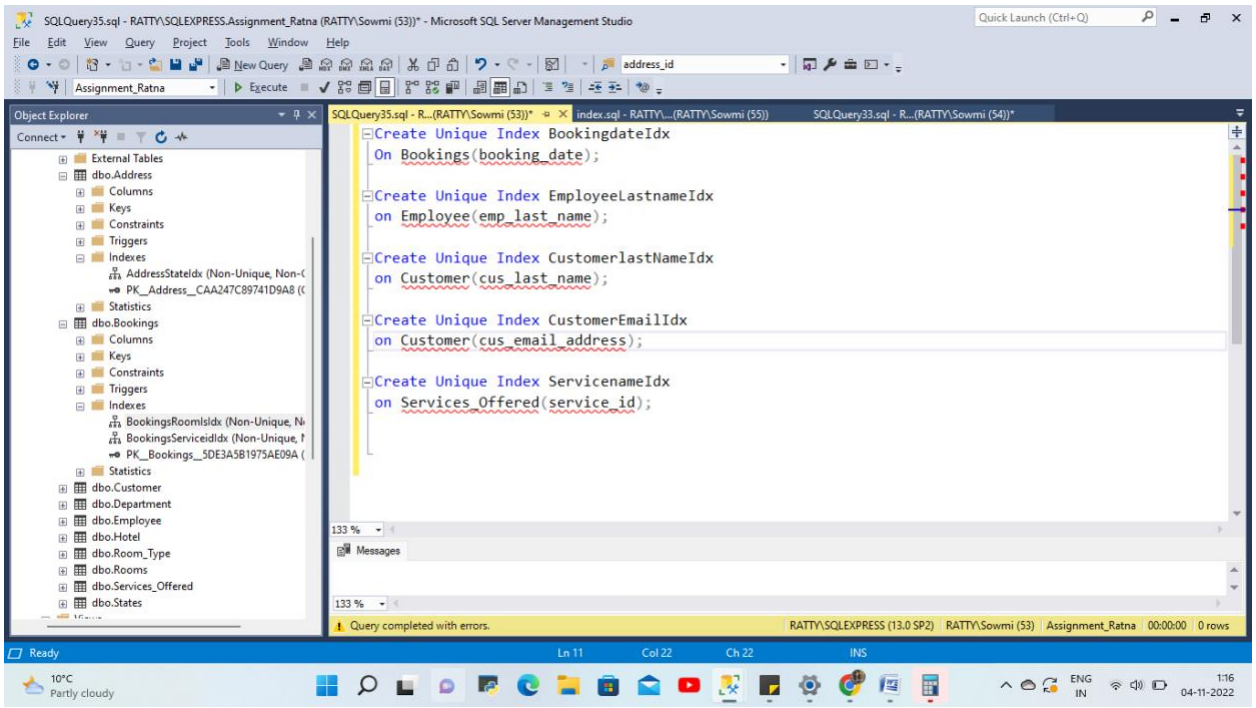
INDEXING DATABASE:

- Primary keys are by default indexed.
  Hotel.hotel_id
  Address.address_id
  State.state_id
  Employee.emp_id
  Customer.cus_id
  Room.room_id
  Room_type.room_type_id
  Bokings.booking_id
  Department.department_id
  Services_offered.service_id

- Foreign keys Columns are to be indexed.



- Some other unique indexes are



Summary and Reflection:

The major users of this application would be the customers and the employees. The customer can login or browse as a guest to check for different type of hotel rooms and different type of services offered. The employees can login and check for the shift timings and the assigned jobs. The employee with manager role can track the employee records and assign the service to be done.

In this iteration, I have created the required tables in the SQL server. Added the Primary key, foreign key and the Unique and Non Unique indexes.

Please fill me in with suggestions and changes or addons that would be required to improve the working of the database.