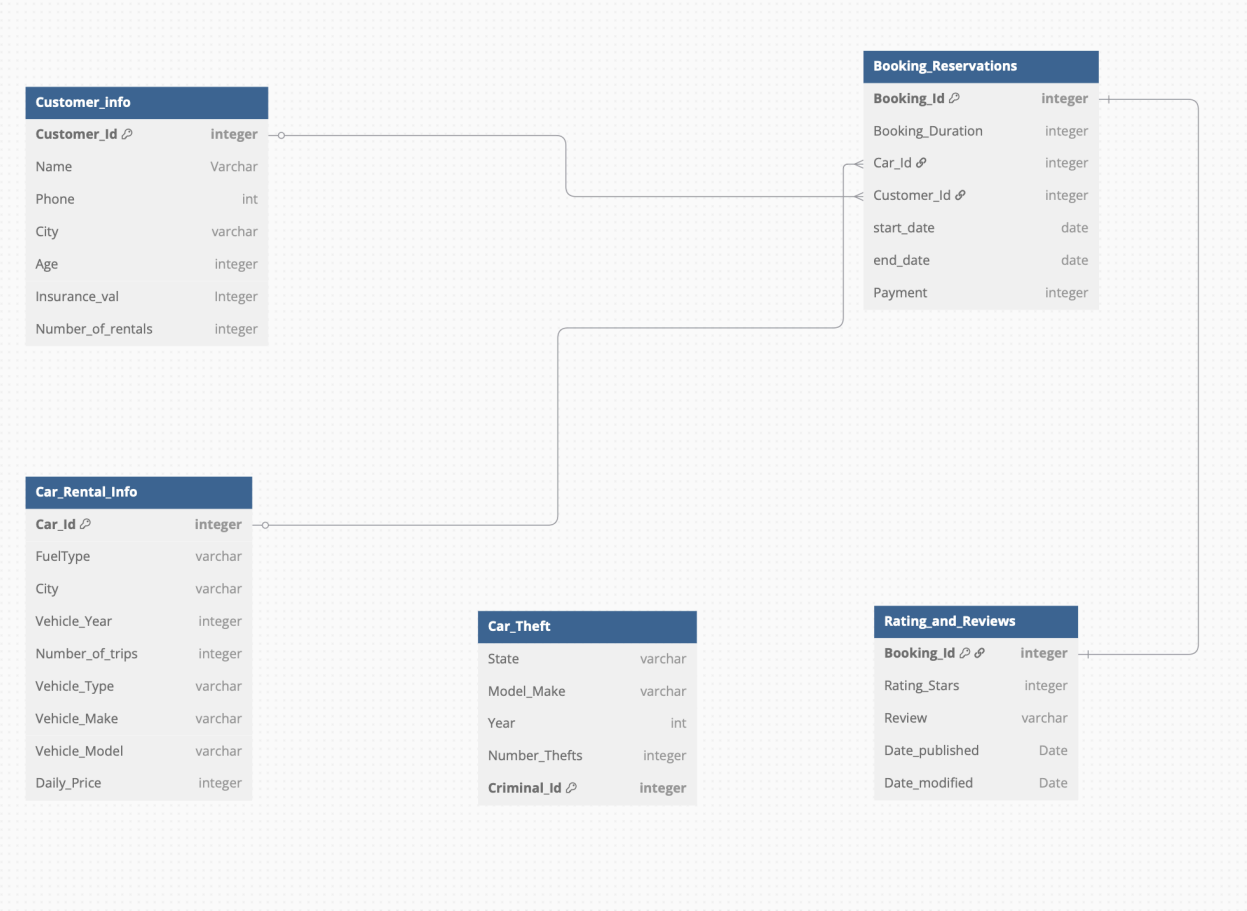
Stage_2:

ER Model:




We have designed the ER model for the project above. The entities, relationships and their assumptions are properly listed below in detail. The annotations on each of the relationships show how we refer to the tables relatively. We can get rid of Car_Id, Customer_Id in Booking_Reservations as the relationships to the table from Car_Rental_Info and Customer_Info inherently show the same. We have kept them for better understanding. Similarly, we have kept the state in the Car_Rental_Info for better understanding.

**Explain your assumptions for each entity and relationship in your model. Discuss why you've modeled something as an entity rather than an attribute of another entity. Describe the cardinality of relationships, like why a student is linked to only one advisor. These assumptions might come from customer requirements or application constraints. Please clarify them.**

1. Customer_Info:
    - Assumptions:
        i. Each customer is uniquely identified by Customer_Id
        ii. Each customer has attributes such as Name, Phone, City, and Age
        iii. Number_of_rentals records the amount of times each customer rented
        iv. Insurance_val depends on Age
    - Explanation:
        i. We model customers as a distinct entity because they have unique identifiers and attributes that are not inherent to any other record in the system.
    - Cardinality of relationships:
        i. A one-to-many relationship “customer_book” to Booking_Reservation represents that a customer is able to have multiple bookings while each booking only belongs to one customer.

2. Booking_Reservations:
    - Assumptions:
        i. Each booking/reservation is uniquely identified by Booking_Id
        ii. A booking/reservation only associate with one customer
        iii. A booking/reservation only involves one car
        iv. Details of the booking/reservation like Booking_Duration, start_date, end_date and Payment are recorded
        v. All attributes naturally depend on Booking_Id
    - Explanation
        i. It is modeled as an entity instead of an attribute in Customer_Info because each customer may have multiple bookings, and each booking has specific details such as price and duration.
    - Cardinality of relationships:
        i. A one-to-one relationship “ratings” to Rating_and_Reviews represents that each booking has only one review and each review is only about one booking

3. Car_Rental_Info
    - Assumptions:
        i. Each rental car is identified by Car_Id

ii. Each car has attributes like FuelType, State, City, Vehicle_Year, Vehicle_Type, Vehicle_Make, Vehicle_Model, Number_of_trips, and Daily_Price
iii. All attributes naturally depend on Car_Id.
iv. We assume that we have at least one car in each state.

- Explanation:
  i. It is a separate entity because cars exist independently of rentals and can be booked multiple times
- Cardinality of relationships:
  i. A one-to-many relationship “car book” to Booking_Reservations represents that a car can be rented multiple times while each booking involves only one car

4. Rating_and_Reviews
   - Assumptions:
     i. Each review is uniquely identified by Booking_Id, making sure that each booking only has one review
     ii. The review has attributes like Rating_Stars, Review, Date_published, and Date_modified
     iii. All attributes naturally depend on Booking_Id
   - Explanation:
     i. It is a separate entity because it maintains user feedback independently of booking details.

5. Car_Theft

   - Assumptions:
     i. Each theft record is uniquely identified by Criminal_Id.
     ii. Theft record are related to car attributes like State, Model, and Number_Thefts
     iii. As Number_Thefts will be recorded/updated annually, it depends on Year
   - Explanation:
     i. It is modeled as a separate entity because theft data is used to adjust insurance rates dynamically

**All the relationships and their cardinality have been explained in detail within the tables.**

**Normalization Form:**

We will be working on the normalization of the above table. We will use the 3NF form. We need to make sure that each table fulfils the criterias of the 3NF from, that is
X -> A is okay if X is a superkey or A is part of a key.

We need to explain and walkthrough this for each of the given tables:

1**.Customer_info** : We can see here that Customer_Id is the primary key of the table. We have two functional dependency, i.e. Customer_Id -> Name, Phone, City, Age, Number_of_Rentals, Insurance_val and Age-> Insurance_val.

As we will use the 3NF form for the normalization, for that we can clearly see that in the functional dependency that we have, Customer_Id is a super key but we have additional dependency Age-> Insurance_val. We have seen that Insurance_val is not part of the candidate key. So, we need to split the table into Customer_info(Customer_Id, Name, Phone, City, Age, Number_of_Rentals) and Insurance_Detail(Age, Insurance_val)

So, our table Customer_info has been modified to be in 3NF form.

2. **Booking_Reservations**: We can see that Booking_Id is the primary key. We have only one functional dependency, i.e. Booking_Id -> Booking_Duration, Customer_Id, Car_Id, Payment, start_date, end_date. We can clearly see that Booking_Id is a superkey, so the table is already in 3NF.

So, our table Booking_Reservations is already in 3NF form.

3. **Car_Rental_Info**: We can see that Car_Id is the primary key and all the other elements of the Table Car_Rental_Info depend only on the Car_Id. We have only one functional dependency, i.e. Car_Id ->FuelType, State, City, Vehicle Year, Number_of_trips, Vehicle_Type, Vehicle_Make, Vehicle_Model, Daily_Price. As we will use the 3NF form for the normalization, for that we can clearly see that in the only functional dependency that we have, Car_Id is a super key.

So, our table Car_Rental_Info  is already in 3NF form.

4.  **Rating_and_Reviews**: We can see that Booking_Id is the primary key and all the other elements of the Table Rating_and_Reviews depend only on the Booking_Id. We have only one functional dependency, i.e. Booking_Id  -> Rating_Stars, Review, Date_published, Date_modified. As we will use the 3NF form for the normalization, for that we can clearly see that in the only functional dependency that we have, Booking_Id is a super key.

So, our table Rating_and_Reviews is already in 3NF form.

5. **Car_Theft**: We have made the Criminal_Id as the primary key for the table. We have only one functional dependency in this table Criminal_Id  -> State, Model_Make, Year, Number_Thefts. As we will use the 3NF form for the normalization, for that we can clearly see that in the only functional dependency that we have, Criminal_Id  is a super key.

So, our table Car_Theft is already in 3NF form.

**Relational Schema:**

We have made a relational schema after the normalization we have done, we will have 6 tables.

**Customer_Info:**

Customer_Info (
    Customer_Id: INT [PK] [FK to Booking_Reservations.Customer_Id]
    Name: VARCHAR(255),
    Phone: INT,
    City: VARCHAR(255),
    Age: INT,
    Number_of_rentals: INT
)

**Insurance_Detail:**

Insurance_Detail(
    Age: INT [PK],
    Insurance_Val: INT
)

**Booking_Reservations:**

Booking_Reservations (
    Booking_Id: INT [PK], [FK to Rating_and_Reviews.Booking_Id],
    Booking_Duration: INT,
    Car_Id: INT,
    Customer_Id: INT,
    Start_date: INT,
    End_date: INT,
    Payment: INT
)

**Car_Rental_Info:**

Car_Rental_Info (
    Car_Id: INT [PK][FK to Booking_Reservations.Car_Id],
    FuelType: VARCHAR(255),
    State: VARCHAR(255),
    City: VARCHAR(255),
    Vehicle_Year: INT,
    Number_of_trips: INT,
    Vehicle_Type: VARCHAR(255),
    Vehicle_Make: VARCHAR(255),
    Vehicle_Model: VARCHAR(255),
    Daily_Price: INT
)

**Car_Theft:**

Car_Theft (
    Criminal_Id: INT [PK],
    State: VARCHAR(255),
    Model_Make: VARCHAR(255),
    Year: INT,
    Number_Thefts: INT
)

**Rating_and_Reviews:**

Rating_and_Reviews (
    Booking_Id: INT [PK]
    Rating_Stars: INT,
    Review: VARCHAR(255),
    Date_published: DATE,
    Date_modified: DATE
)

**Summary**

Our database design for the car rental system includes a comprehensive ER model with five key entities: **Customer_Info, Booking_Reservations, Car_Rental_Info, Car_Theft, and Rating_and_Reviews**.

The ER model establishes important relationships: customers have one-to-many relationships with bookings, cars have one-to-many relationships with bookings, bookings have one-to-one relationships with reviews, and car rental information connects to theft data through states.

We documented specific assumptions for each of the following: Customer_Info is uniquely identified by Customer_Id with attributes such as Name, Phone, City, Age, and Number_of_rentals; Booking_Reservations uses Booking_Id as its primary key and records details such as Booking_Duration and Payment; Car_Rental_Info uses Car_Id with attributes such as FuelType, State, and Daily_Price; Car_Theft records are identified by Criminal_Id and track theft data by State, Model, and Year; and Rating_and_Reviews uses Booking_Id as its primary key and includes Rating_Stars and Review attributes. To ensure proper functional dependencies, we applied 3NF normalization, splitting Customer_Info into Customer_Info and Insurance_Detail tables because Insurance_val depends on Age rather than the primary key; and all other tables (Booking_Reservations, Car_Rental_Info, Rating_and_Reviews, and Car_Theft) were already in 3NF form with the necessary functional dependencies.

The final schema has six tables with clearly defined primary and foreign keys. According to our assumptions, we have made all the required relationships and entities.