CSCI 5408

DATA MANAGEMENT AND WAREHOUSING

LAB ASSIGNMENT - 2

Banner ID: B00948977

Git Assignment Link :
https://git.cs.dal.ca/sukumaran/csci5408_f23_b00948977_balaji_sukumaran/-/tree/main/Lab2

Table of contents

Problem Statement 1: Design an ERD/ EERD for Airbnb hotel **1**

1.1: Identify the entities and attributes (Minimum 8 entities) **1**

1.2: Design a basic conceptual, logical, physical model **2**

1.3: Create an ERD in MySQL workbench using forward engineering **12**

Problem Statement 1: Design an ERD/ EERD for Airbnb hotel

1.1: Identify the entities and attributes (Minimum 8 entities): Following are the main entities and attributes for the Airbnb hotel.

Note: relationship between tables will be established in further steps.

USER
+ user_id : **PK**
+ fname
+ lname
+ phoneno
+ aboutme

STAY
+ stay_id: **PK**
+ landmark
+ type
+ address

BOOKING
+ booking_id : **PK**
+ created_on
+ modified_on
+ from_date
+ to_date

OWNER
+ ssn : **PK**
+ fname
+ lname
+ address

PAYMENT
+ payment_id : **PK**
+ payment_mode
+ rent

HOST
+ ssn : **PK**
+ fname
+ lname
+ address
+ salary
s

GUEST
+ fname
+ lname
+ relationship
+ age

AMENITIES
+ amenity_id : **PK**
+ name
+ charges
+ age_limit

SUPPORT
+ ticket_id: **PK**
+ maintenance_id: **PK**
+ support_fname
+ support_lname
+ maintenance_staff_fname
+ maintenance_staff_lname
+ service
+ service_charge

Problem Statement 2: **Design a basic conceptual, logical, physical model**

2.1: Conceptual Model : The conceptual model for Airbnb has been build using the following assumptions:

- N User makes 1 Booking (group of users makes a booking)
- 1 User pays for 1 booking (1 person among the group pays rent)
- 1 User pays in N Payments for 1 Booking (pay in installments for long term stay)
- Each User can bring N guests
- 1 User raises N support tickets
- N Support services on 1 Stay
- 1 Stay is owned by N Owners
- 1 Stay is hosted by 1 host
- 1 stay has N Amenities

Chen-Model:

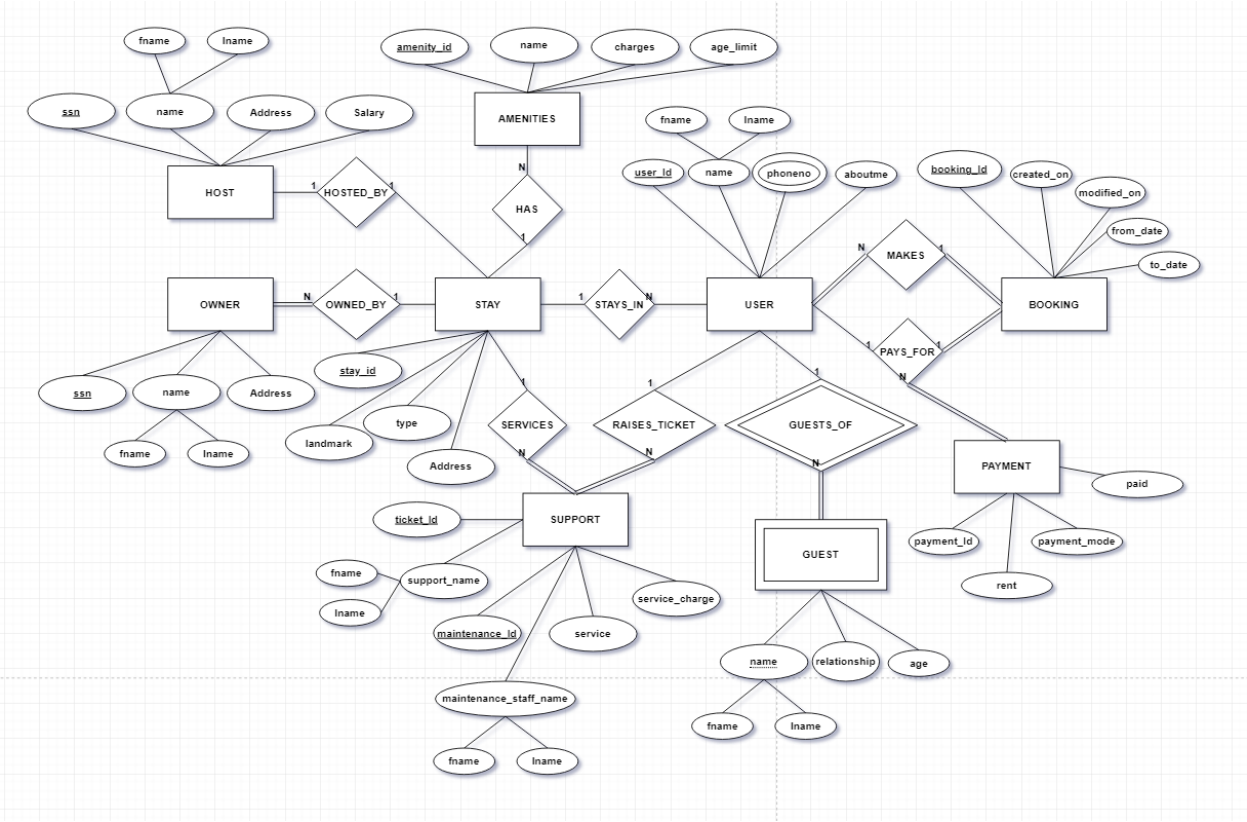

Figure 1: Chen-Model Airbnb

Crow-Foot Model:

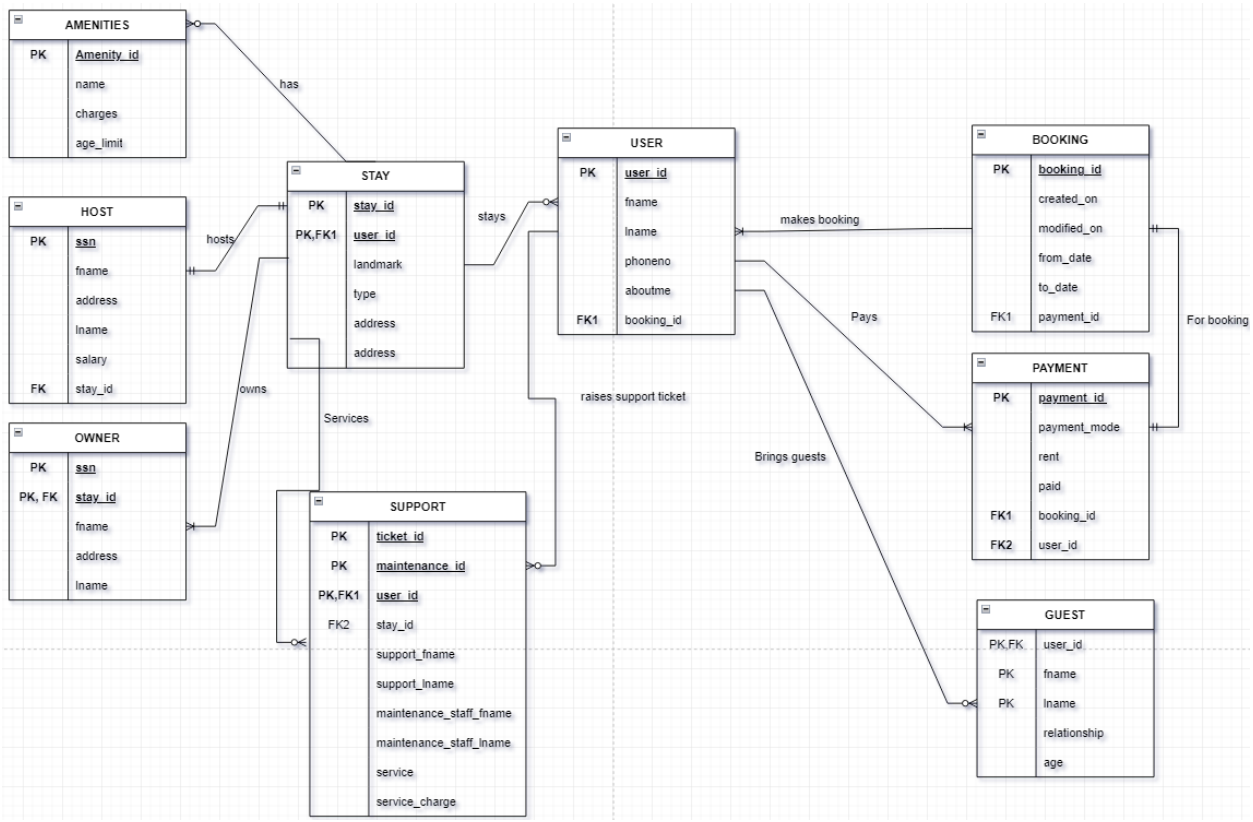

Figure 2: Crow-model Airbnb

Design Issues:

1. **Fan Trap:** Here the Fan Trap issue exists between User, Stay and Amenities, as Stay is in multiple 1:M relationship and is not consistent with real world

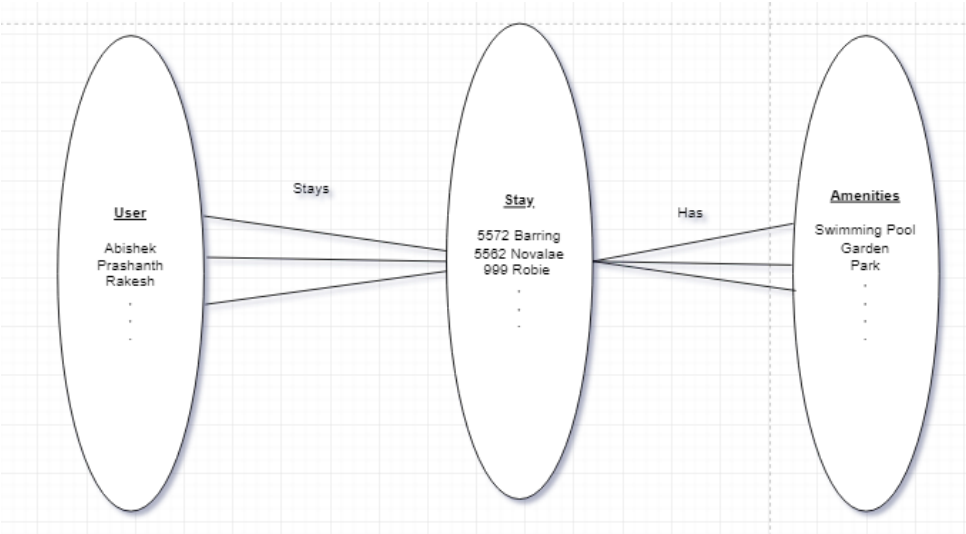

Figure 3: Fan-Trap between user, stay, and amenities

This situation can be solved by establishing 1:M relationship between Stay and User, and 1:M relationship between User and Amenities.

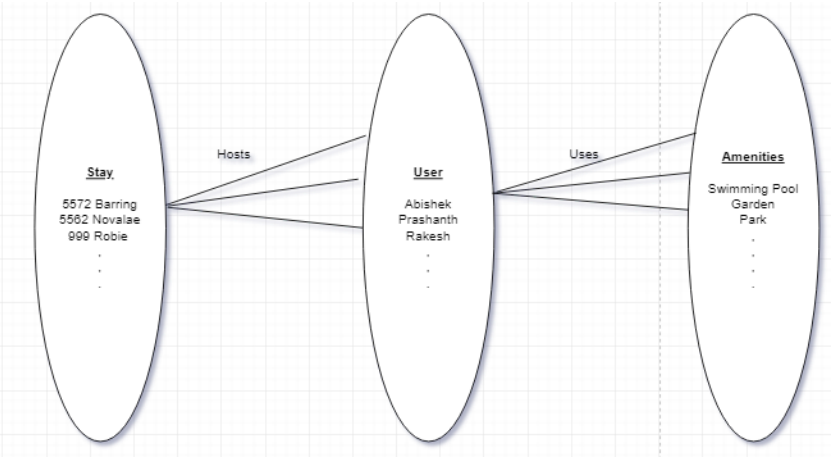

Figure 4: Fan-Trap solution for stay, user and amenities

2. **Chasm Trap:** Model suggest there's a relationship between the owner and user. But it's not the case, because it is the host's responsibility to host the user. Owner and user is NOT related.

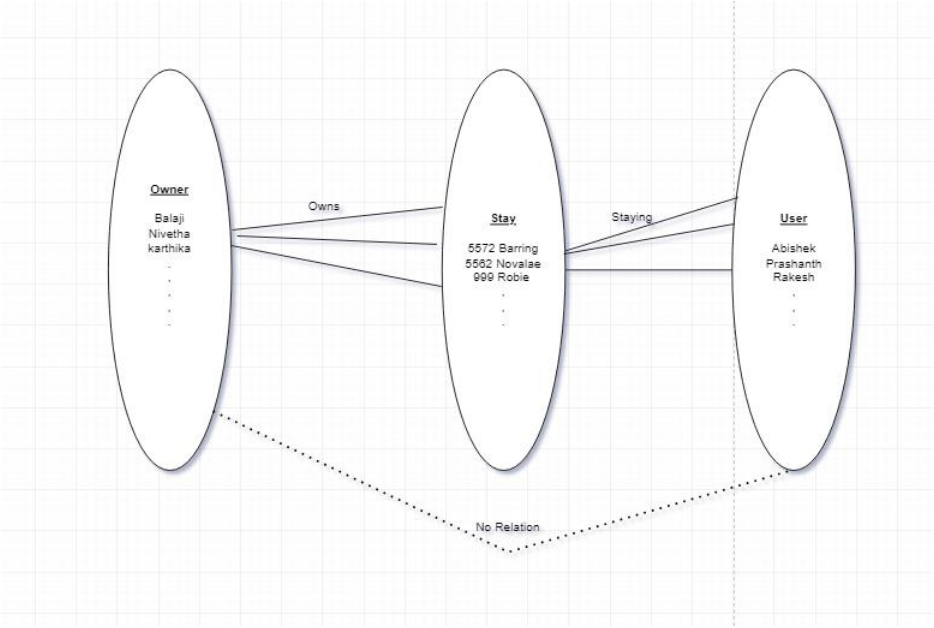

Figure 5: Chasm trap in owner, stay and user

Updated conceptual model after fixing design issue: Now relationship exists between user and amenities

Chen-Model

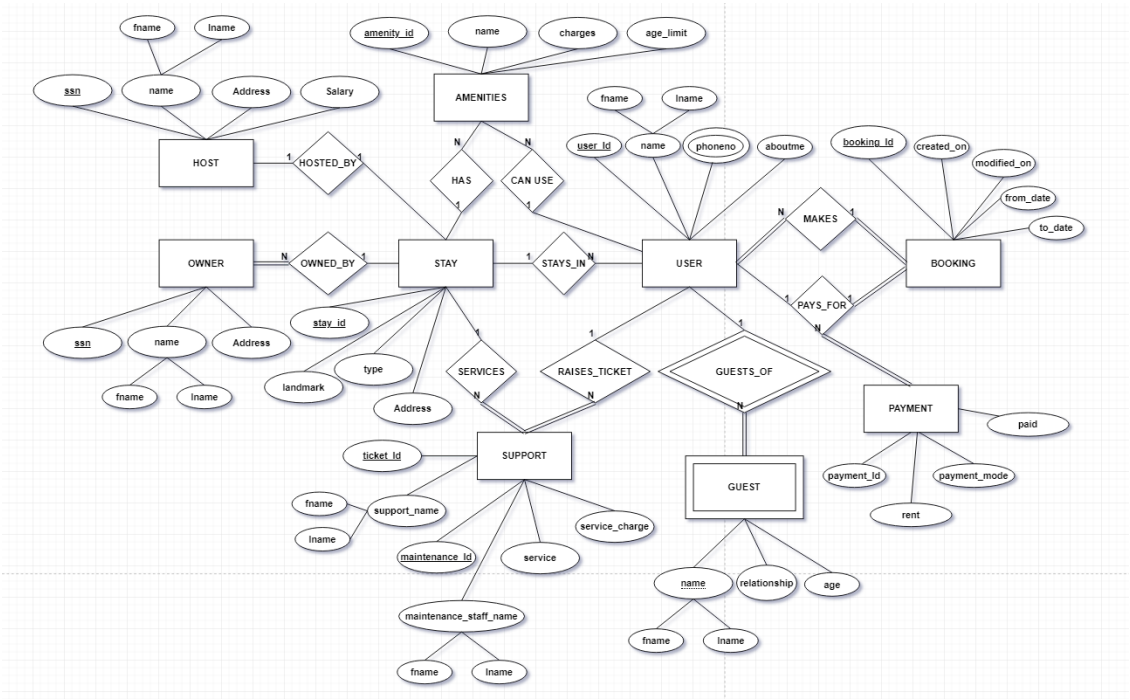

Figure 6: Chen-Model Airbnb post fixing design issues

Crow-Foot Model:

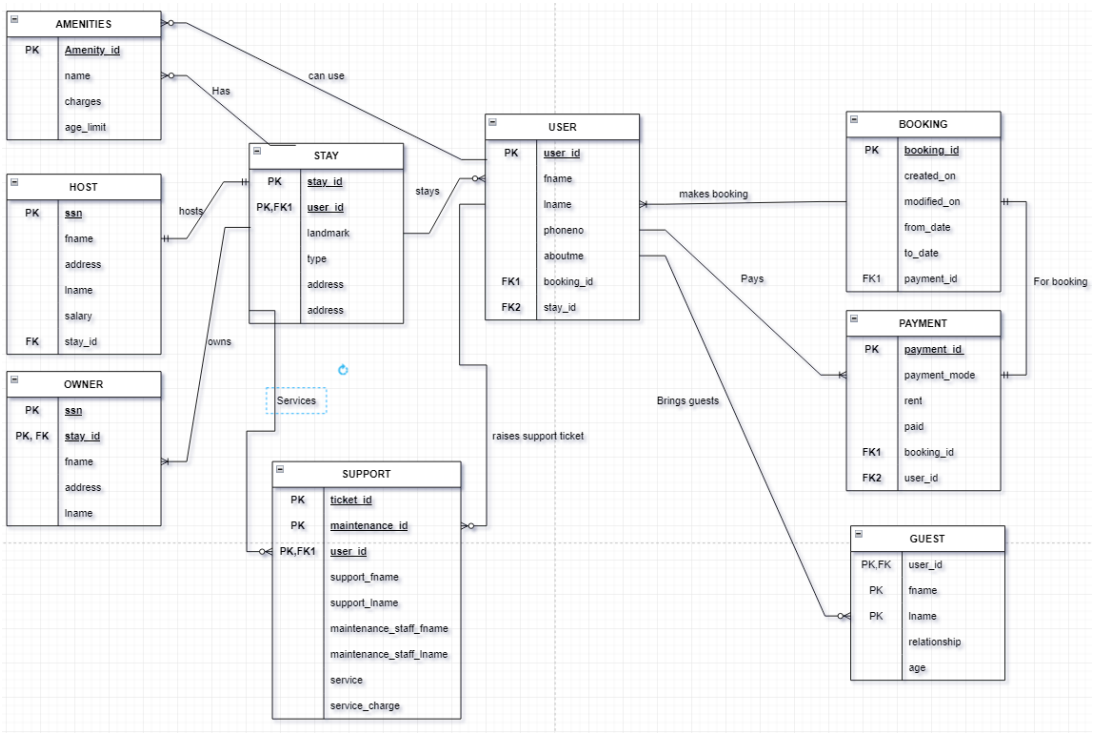

Figure 7: Crow-foot model Airbnb post fixing design issue

2.2: Logical Model : To create a logical model, we have to do the Normalization process:

2.2.1. Normalization

1NF: 1NF is completed when the following achieved:

- All key attributes are defined
- There are no repeating groups in the table
- All attributes are dependent on the primary key

In our model, the following are the key attributes.

Table 1: User, Primary Key {user_id}, Foreign Key {booking_id, stay_id}

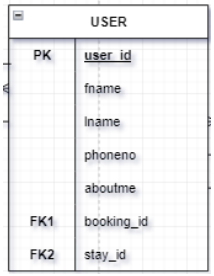

Figure 8: User entity

Table 2: Booking, Primary Key {booking_id}, Foreign Key {payment_id}

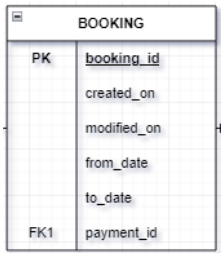

Figure 9: Booking entity

Table 3: Payment, Primary Key { payment_id }, Foreign Key {booking_id, user_id}

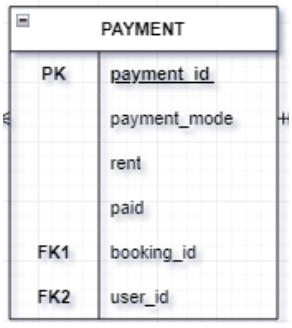

Figure 10: Payment entity

Table 4: Guest, Primary Key { user_id, fname, lname }, Foreign Key {user_id}

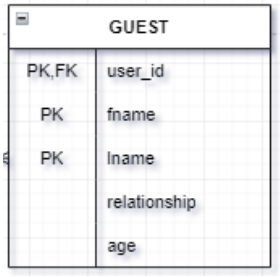

Figure 11: Guest entity

Table 5: Support, Primary Key { ticket_id, maintenance_id, user_id}, Foreign Key {user_id, stay_id}

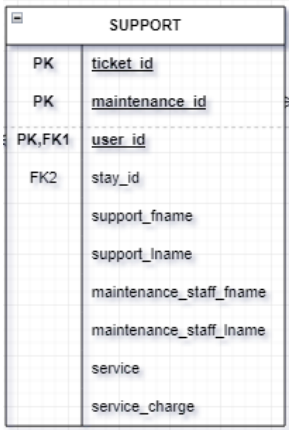

Figure 12: Support entity

Table 6: Owner, Primary Key { ssn, stay_id}, Foreign Key {stay_id}

Figure 13: Owner entity

Table 7: Host, Primary Key { ssn }, Foreign Key {stay_id}

Figure 14: Host entity

Table 8: Amenities, Primary Key {Amenity_id}

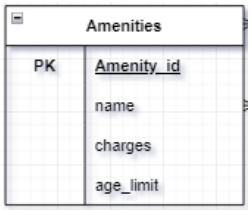

Figure 15: Amenities entity

Table 9: Stay, Primary Key {stay_id, user_id}, Foreign key {user_id}

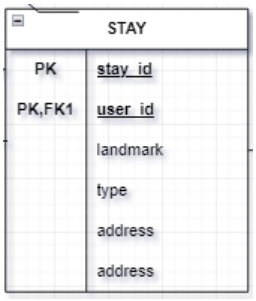

Figure 16: Stay entity

2NF: Partial dependency exists in table Support.

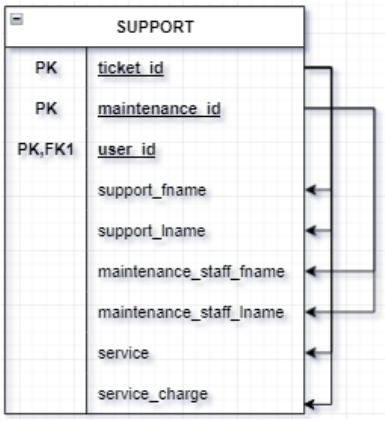

Figure 17: Support entity partial dependency

- support_fname, support_lname identified by ticket_id
- maintenance_staff_fname, maintenance_staff_lname identified by maintenance_id

2 NF can be achieved by, removing partial dependency by spitting the support table in to support and maintenance

Support table: Primary key {ticket_id }, Foreign key {user_id}

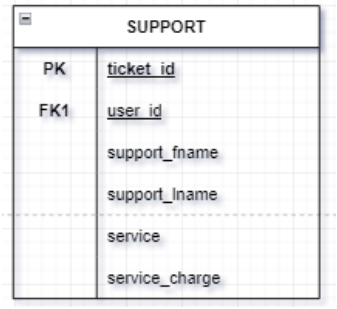

Figure 18: Support entity after eliminating partial dependency

Maintenance table: Primary key {maintenance_id }, Foreign key {ticket_id, stay_id}

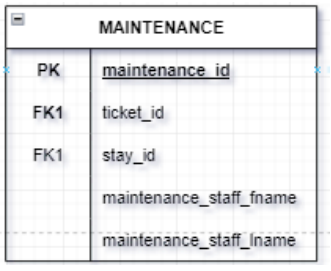

Figure 19: Maintenance entity after eliminating partial dependency

3NF: Transitive dependency exists in table support, because service charge is dependent on the service.

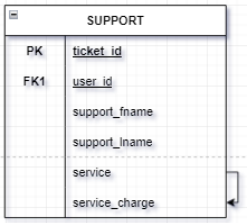

Figure 20: Support entity in transitive dependency

3NF can be achieved by transitive dependency, by decoupling support entity into support and services table

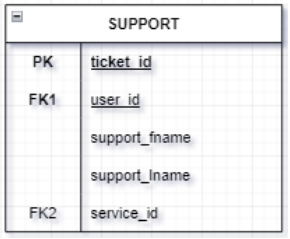

Figure 21: Support entities after removing transitive dependency

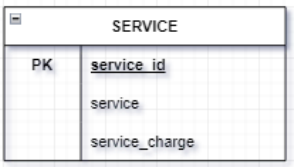

Figure 22: service entities after removing transitive dependency

Crow-Foot model of Airbnb after normalizing

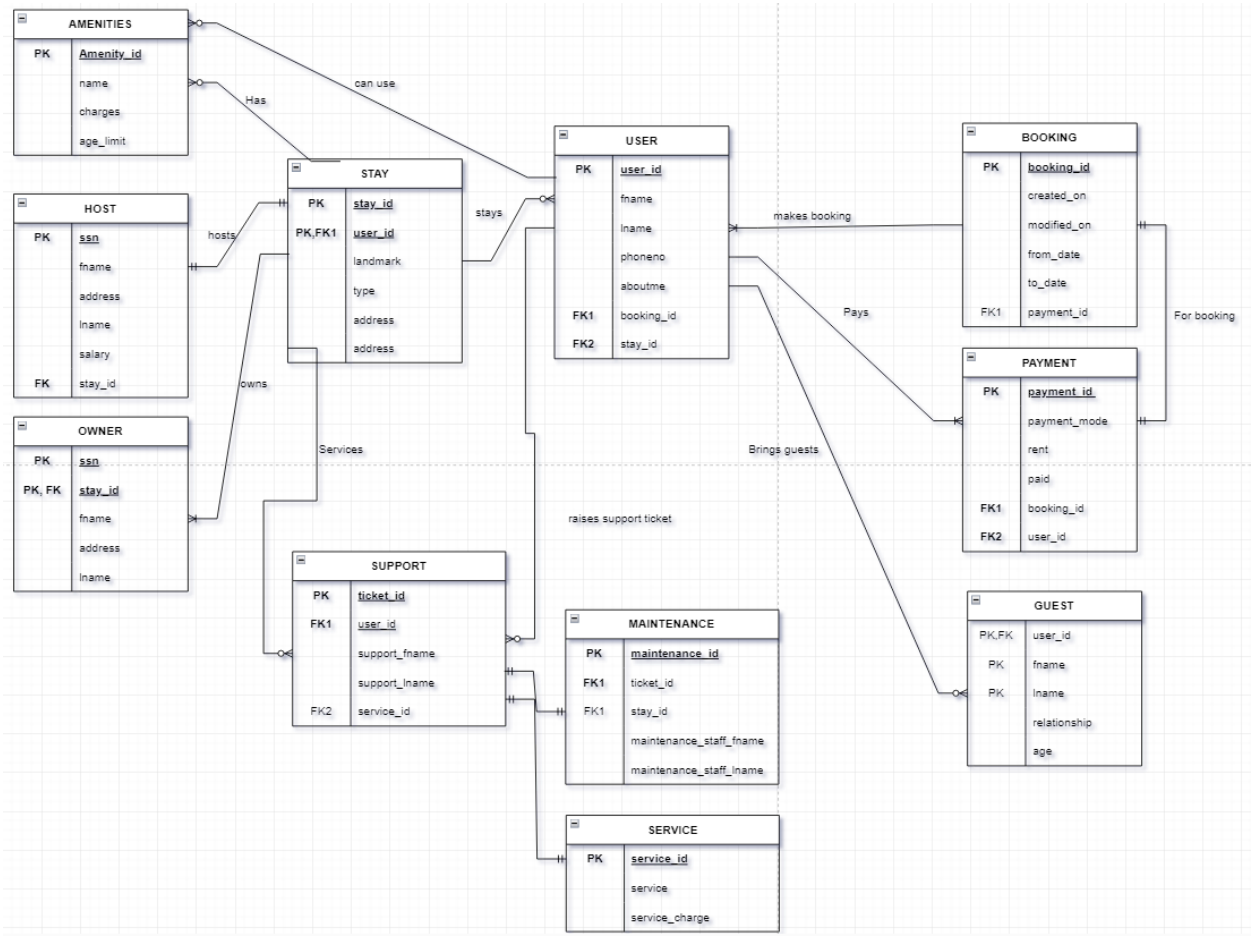

Figure 23: Crow-foot model of Airbnb after normalization

Problem Statement 3: **Create an ERD in MySQL workbench using forward engineering**

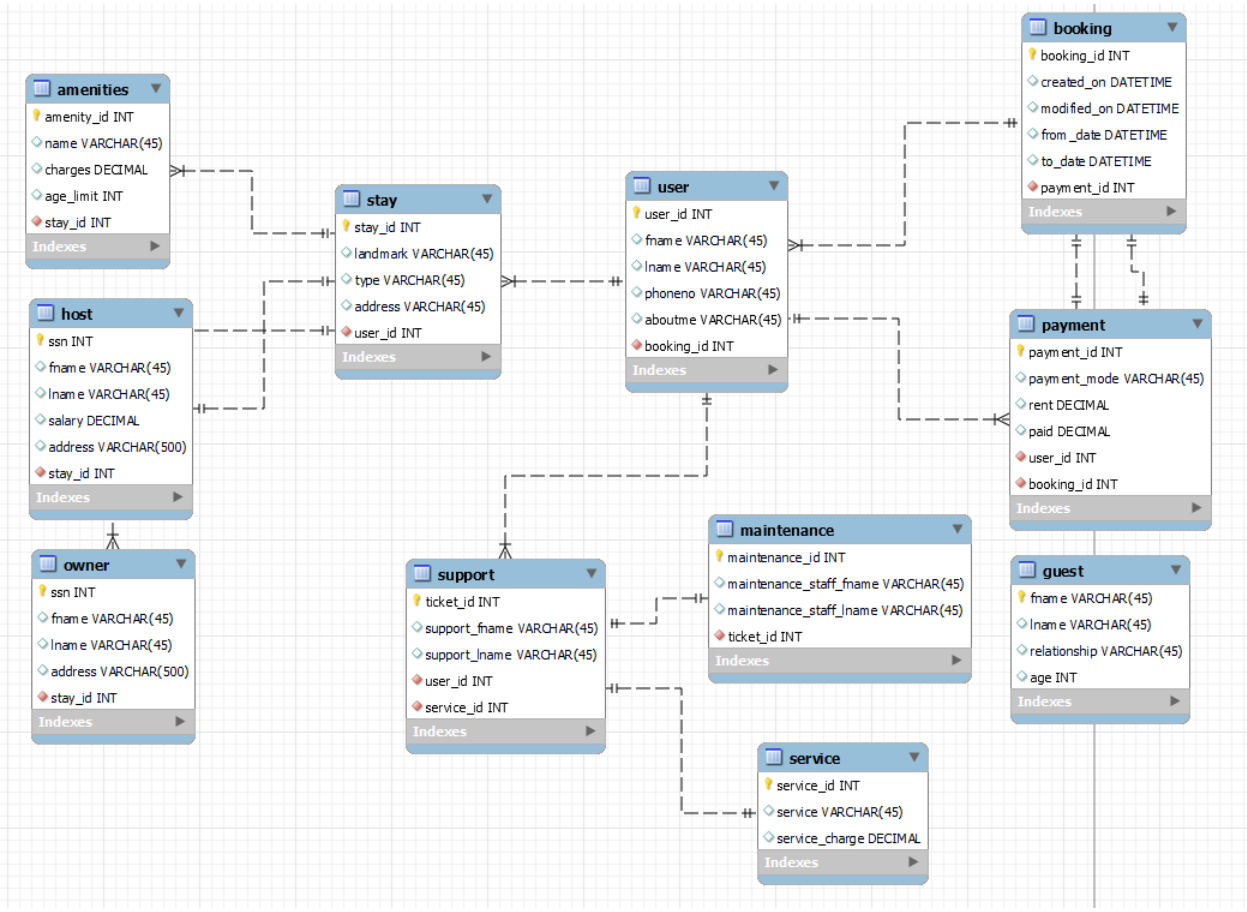

Figure 24: ERD for Airbnb in MySql workbench

Forward-Engineered Query

```
-- MySQL Workbench Forward Engineering

SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,NO_ZERO_DATE,ERROR_FOR_DIVISION_BY_ZERO,NO_ENGINE_SUBSTITUTION';

-- -----------------------------------------------------
-- Schema mydb
-- -----------------------------------------------------

-- -----------------------------------------------------
-- Schema mydb
-- -----------------------------------------------------
CREATE SCHEMA IF NOT EXISTS `mydb` DEFAULT CHARACTER SET utf8 ;
USE `mydb` ;

-- -----------------------------------------------------
-- Table `mydb`.`payment`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`payment` (
  `payment_id` INT NOT NULL,
  `payment_mode` VARCHAR(45) NULL,
  `rent` DECIMAL NULL,
  `paid` DECIMAL NULL,
  `user_id` INT NOT NULL,
  `booking_id` INT NOT NULL,
  PRIMARY KEY (`payment_id`),
  INDEX `fk_payment_user1_idx` (`user_id` ASC) VISIBLE,
  INDEX `fk_payment_booking1_idx` (`booking_id` ASC) VISIBLE,
  CONSTRAINT `fk_payment_user1`
    FOREIGN KEY (`user_id`)
    REFERENCES `mydb`.`user` (`user_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_payment_booking1`
    FOREIGN KEY (`booking_id`)
    REFERENCES `mydb`.`booking` (`booking_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `mydb`.`booking`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`booking` (
  `booking_id` INT NOT NULL,
  `created_on` DATETIME NULL,
  `modified_on` DATETIME NULL,
  `from_date` DATETIME NULL,
  `to_date` DATETIME NULL,
  `payment_id` INT NOT NULL,
  PRIMARY KEY (`booking_id`),
  INDEX `fk_booking_payment1_idx` (`payment_id` ASC) VISIBLE,
  CONSTRAINT `fk_booking_payment1`
    FOREIGN KEY (`payment_id`)
    REFERENCES `mydb`.`payment` (`payment_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `mydb`.`user`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`user` (
  `user_id` INT NOT NULL,
  `fname` VARCHAR(45) NULL,
  `lname` VARCHAR(45) NULL,
```

```
    `phoneno` VARCHAR(45) NULL,
    `aboutme` VARCHAR(45) NULL,
    `booking_id` INT NOT NULL,
    PRIMARY KEY (`user_id`),
    INDEX `fk_user_booking_idx` (`booking_id` ASC) VISIBLE,
    CONSTRAINT `fk_user_booking`
      FOREIGN KEY (`booking_id`)
      REFERENCES `mydb`.`booking` (`booking_id`)
      ON DELETE NO ACTION
      ON UPDATE NO ACTION)
  ENGINE = InnoDB;


  -- -----------------------------------------------------
  -- Table `mydb`.`guest`
  -- -----------------------------------------------------
  CREATE TABLE IF NOT EXISTS `mydb`.`guest` (
    `fname` VARCHAR(45) NOT NULL,
    `lname` VARCHAR(45) NULL,
    `relationship` VARCHAR(45) NULL,
    `age` INT NULL,
    PRIMARY KEY (`fname`))
  ENGINE = InnoDB;


  -- -----------------------------------------------------
  -- Table `mydb`.`service`
  -- -----------------------------------------------------
  CREATE TABLE IF NOT EXISTS `mydb`.`service` (
    `service_id` INT NOT NULL,
    `service` VARCHAR(45) NULL,
    `service_charge` DECIMAL NULL,
    PRIMARY KEY (`service_id`))
  ENGINE = InnoDB;
```

```
104
105
106     -- -----------------------------------------------------
107     -- Table `mydb`.`support`
108     -- -----------------------------------------------------
109 •   CREATE TABLE IF NOT EXISTS `mydb`.`support` (
110       `ticket_id` INT NOT NULL,
111       `support_fname` VARCHAR(45) NULL,
112       `support_lname` VARCHAR(45) NULL,
113       `user_id` INT NOT NULL,
114       `service_id` INT NOT NULL,
115       PRIMARY KEY (`ticket_id`),
116       INDEX `fk_support_user1_idx` (`user_id` ASC) VISIBLE,
117       INDEX `fk_support_service1_idx` (`service_id` ASC) VISIBLE,
118       CONSTRAINT `fk_support_user1`
119         FOREIGN KEY (`user_id`)
120         REFERENCES `mydb`.`user` (`user_id`)
121         ON DELETE NO ACTION
122         ON UPDATE NO ACTION,
123       CONSTRAINT `fk_support_service1`
124         FOREIGN KEY (`service_id`)
125         REFERENCES `mydb`.`service` (`service_id`)
126         ON DELETE NO ACTION
127         ON UPDATE NO ACTION)
128     ENGINE = InnoDB;
129
130
131     -- -----------------------------------------------------
132     -- Table `mydb`.`maintenance`
133     -- -----------------------------------------------------
134 •   CREATE TABLE IF NOT EXISTS `mydb`.`maintenance` (
135       `maintenance_id` INT NOT NULL,
136       `maintenance_staff_fname` VARCHAR(45) NULL,
137       `maintenance_staff_lname` VARCHAR(45) NULL,
138       `ticket_id` INT NOT NULL,
139       PRIMARY KEY (`maintenance_id`),
140       INDEX `fk_maintenance_support1_idx` (`ticket_id` ASC) VISIBLE,
141       CONSTRAINT `fk_maintenance_support1`
142         FOREIGN KEY (`ticket_id`)
143         REFERENCES `mydb`.`support` (`ticket_id`)
144         ON DELETE NO ACTION
145         ON UPDATE NO ACTION)
146     ENGINE = InnoDB;
147
148
149     -- -----------------------------------------------------
150     -- Table `mydb`.`stay`
151     -- -----------------------------------------------------
152 •   CREATE TABLE IF NOT EXISTS `mydb`.`stay` (
153       `stay_id` INT NOT NULL,
154       `landmark` VARCHAR(45) NULL,
155       `type` VARCHAR(45) NULL,
156       `address` VARCHAR(45) NULL,
157       `user_id` INT NOT NULL,
158       PRIMARY KEY (`stay_id`),
159       INDEX `fk_stay_user1_idx` (`user_id` ASC) VISIBLE,
160       CONSTRAINT `fk_stay_user1`
161         FOREIGN KEY (`user_id`)
162         REFERENCES `mydb`.`user` (`user_id`)
163         ON DELETE NO ACTION
164         ON UPDATE NO ACTION)
165     ENGINE = InnoDB;
166
167
168     -- -----------------------------------------------------
169     -- Table `mydb`.`amenities`
170     -- -----------------------------------------------------
171 •   CREATE TABLE IF NOT EXISTS `mydb`.`amenities` (
172       `amenity_id` INT NOT NULL,
173       `name` VARCHAR(45) NULL,
174       `charges` DECIMAL NULL,
175       `age_limit` INT NULL,
176       `stay_id` INT NOT NULL,
177       PRIMARY KEY (`amenity_id`),
178       INDEX `fk_amenities_stay1_idx` (`stay_id` ASC) VISIBLE,
179       CONSTRAINT `fk_amenities_stay1`
180         FOREIGN KEY (`stay_id`)
181         REFERENCES `mydb`.`stay` (`stay_id`)
182         ON DELETE NO ACTION
183         ON UPDATE NO ACTION)
184     ENGINE = InnoDB;
```

```
-- -----------------------------------------------------
-- Table `mydb`.`host`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`host` (
  `ssn` INT NOT NULL,
  `fname` VARCHAR(45) NULL,
  `lname` VARCHAR(45) NULL,
  `salary` DECIMAL NULL,
  `address` VARCHAR(500) NULL,
  `stay_id` INT NOT NULL,
  PRIMARY KEY (`ssn`),
  INDEX `fk_host_stay1_idx` (`stay_id` ASC) VISIBLE,
  CONSTRAINT `fk_host_stay1`
    FOREIGN KEY (`stay_id`)
    REFERENCES `mydb`.`stay` (`stay_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `mydb`.`owner`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`owner` (
  `ssn` INT NOT NULL,
  `fname` VARCHAR(45) NULL,
  `lname` VARCHAR(45) NULL,
  `address` VARCHAR(500) NULL,
  `stay_id` INT NOT NULL,
  PRIMARY KEY (`ssn`),
  INDEX `fk_owner_stay1_idx` (`stay_id` ASC) VISIBLE,
  CONSTRAINT `fk_owner_stay1`
    FOREIGN KEY (`stay_id`)
    REFERENCES `mydb`.`stay` (`stay_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```