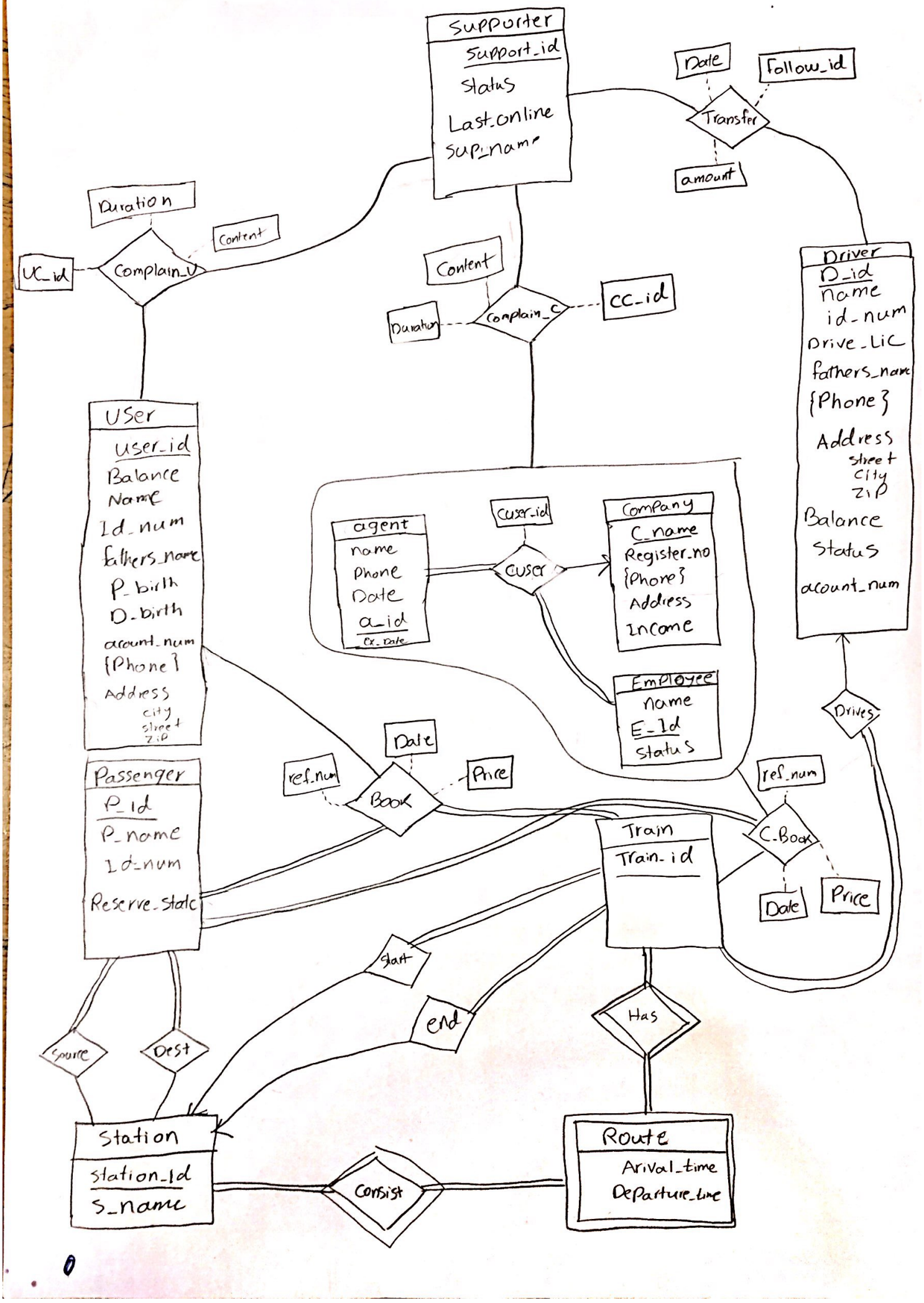

Passenger=(P_id , Source, Dest, P_name, Id_num, Reserve_state)

User=(user_id , Balance, U_Name, Id_num, fathers_name, P_birth, D_birth

acount_num, City, ZiP, Street)

U_Phone = (user_id, Phone)

Route(Train_id , station_id , Arival_time, Departure_time)

Train=(Train_id, D_id, start, end)

Book_U=(user_id, P_id, Train_id, Date, Price , ref_num)

Book_C = (Cuser_id , P_id , Train_id, Date, Price , ref_num)

agent = (a_id, name, phone, ex_Date

company = (C_name, Register_no, Income, City, street, ZiP,

C_Phone = (C_name, phone)

Driver = (D_id, name, id_num, Drive_lic, fathers_name, City, street,

ZiP, Balance, Status, acount_num)

supporter = (support_id, status, last_online, sup_name

Transfer = (follow_id , support_id, D_id, Date, amount)

Complain_C=(CC_id , Support_id, a_id, E_id, C_name, Duration, Conten

Complain_U = (UC_id, user_id, Support_id, Duration , Content)

Station = (Station_id, S_name)

Employee = (E_id, name, Status)

Cuser(Cuser_id, E_id, a_id, Cnam

BCNF

Passenger:

P_id → Source, Dest, P_name, Id_num, Reserve_stat) ✓

User:

User_id → Balance, U_name, Id_num, fathers_name, P_birth, D_birth
acount_num, City, Zip, Street) ✓

U_Phone:

user_id → Phone ✓

Route:

Train_id → station_id, arrival_time, Departure_time ✓

Train:

Train_id → D_id, capacity ✓

Book_C:

ref_num → Cuser_id, P_id , Train_id, Date, Price ✓

Book_U:

ref_num → User_id, P_id, Train_id, Date, Price ✓

agent:

a_id → name, phone ✓

Company:

C_name → Register_no, Income, City, Street, zip ✓

C_Phone:

C_name → Phone ✓

Driver :

D_id → name, Id_num, Drive_lic, fathers_name, City, street, ZiP, ✓

Supporter:

support_id → status, last_online, sup_name ✓

Transfer :

follow_id → support_id, D_id, Date, amount ✓

Complain_C :

CC_id → support_id, a_id, E_id, C_name, Date ✓

Complain_U:

UC_id → user_id, support_id, Date ✓

Station :

station_id → S_name ✓

Employee :

E_id → name, status ✓

برای گرفتن log سیستم از trigger استفاده شده است. که یک مثال از این trigger در زیر آورده شده است. تمامی کد های مربوط به trigger های جدول های دیگر در فایل train.sql آورده شده است.

همانطور که مشاهده می‌شود برای هر عمل insert و update و delete یک trigger جدا نوشته شده است. Insert trigger بعد از عمل insert فعال می‌شود تا سطر جدید اضافه شده را در جدول log ذخیره کند برای update trigger نیز همین طور است. اما برای delete trigger قبل از پاک کردن سطر مورد نظر این سطر را داخل جدول log ذخیره می‌کنیم.

```
--
-- Triggers `book_u`
--
DELIMITER $$
CREATE TRIGGER `delete_book_u` BEFORE DELETE ON `book_u` FOR EACH ROW BEGIN
        INSERT INTO book_u_log(action, ref_num, P_id, Train_id, User_id, Date, price,
create_date, modify_date)

VALUES('delete',OLD.ref_num,OLD.P_id,OLD.Train_id,OLD.User_id,OLD.Date,OLD.price,OLD.c
reate_date, OLD.modify_date);
END
$$
DELIMITER ;
DELIMITER $$
CREATE TRIGGER `insert_book_u` AFTER INSERT ON `book_u` FOR EACH ROW BEGIN
        INSERT INTO book_u_log(action, ref_num, P_id, Train_id, User_id, Date, price,
create_date, modify_date)

VALUES('insert',NEW.ref_num,NEW.P_id,NEW.Train_id,NEW.User_id,NEW.Date,NEW.price,NEW.c
reate_date, NEW.modify_date);
END
$$
DELIMITER ;
DELIMITER $$
CREATE TRIGGER `update_book_u` AFTER UPDATE ON `book_u` FOR EACH ROW BEGIN
        INSERT INTO book_u_log(action, ref_num, P_id, Train_id, User_id, Date, price,
create_date, modify_date)
```

```
VALUES('update',NEW.ref_num,NEW.P_id,NEW.Train_id,NEW.User_id,NEW.Date,NEW.price,NEW.c
reate_date, NEW.modify_date);

END

$$

DELIMITER ;
```