We affirm that we have carried out my academic endeavors with full academic honesty. We understand that by attaching this pledge to the first piece of work we hand in, we are making the same pledge for all subsequent work in the course.

Signed. Khai Dong, Tuan Trinh.

Update to the original design report is in blue

Group Member: Khai Dong, Tuan Trinh

RENTAL APARTMENT DATABASE

I. Design Report

(1) The LAND company owns an apartment complex consisting of 15 buildings that accommodate 900 tenants and employ 150 employees. Each building is identified by its unique building number which is assigned upon the purchase of the complex. For each building, besides the building number, the company wants to keep track of the building's build date, the number of floors and rooms in the building, the building's address, and whether the building allows pets.

(2) Each building employs 10-15 employees depending on the size of the building. Each employee has an SSN, name (first name, last name, and middle initials), sex, date of birth, full address, email address (which is uniquely generated on the hired date), phone numbers, the hire date, and hourly rate (in dollars). Employees' phones are assumed to not be shared with any employee. This means a phone is unique to an employee, and there could be several phones that imply the same employee.

(3) There are 2 types of employees: managers and workers. A manager must manage a particular building, and a building can only have 1 manager. A worker is assigned to a particular building. A building can have multiple workers working. The LAND company also wants to keep track of the type of worker (janitor, plumber, electrician, security, etc.,)

(4) Each building must have an apartment and can have up to 20 apartments. Each apartment has a room number (e.g., room 01) and floor (e.g., floor 2). Note that the floor number is not unique between buildings and the room number is not unique between floors. The company

also wants to keep track of the number of bathrooms and bedrooms, the apartment's type (duplex, normal. penthouse), and the size (in squared feet) of each apartment.

(5) In each building, there are several services offered (e.g., laundry, gym, swimming pool, etc.,). By the LAND company's policies, for these services to run within the buildings, they must open daily. The LAND company wants to keep track of the name of the service (assumed to be unique within the same building), the daily opening and closing time of the service, the type of service, and the name of the service's operating company.

(6) Each building has a unique parking lot. The parking lot shares the address with its building. For each parking lot, the company wants to keep track of the parking lot number (assigned by the company and unique for each parking lot), and the capacity of the parking lot. For each parking lot, there are multiple parking slots. Each parking slot has a section number and a slot number. Noted that the section number is only unique within the parking lot and the slot number is only unique within the section. For each parking slot, we also want to keep track of which type of vehicle which this particular slot is for (e.g., mopeds, bikes, cars, etc.,)

(7) The LAND company also keeps track of people attending to the services or living in the company's property. Each person has a name (first name, last name, middle initials), sex, date of birth, age, phone, and email address (assumed to be unique to a person). Each person is either a tenant who is living in the company's apartment or a non-tenant subscribing to the services provided in buildings. For a non-tenant, the company further keeps track of their address. For a tenant, for credit and background checks, the company keeps track of their SSN. A tenant may also have dependents that live in the apartment with them. For security purposes, it is important to not keep track of everyone's SSN if not specifically required; this means the company asked that the database strictly only keeps track of tenants' SSNs to

reduce possible legal implications. A person’s phone number is assumed to be unique and is used for identification upon entering the buildings and subscribing to services offered in the buildings.

(8) Each person may subscribe to services provided within the buildings. For each person currently subscribing to a service, the company wants to keep track of the subscription's start date, end date, and type (e.g., personal, business, etc.,).

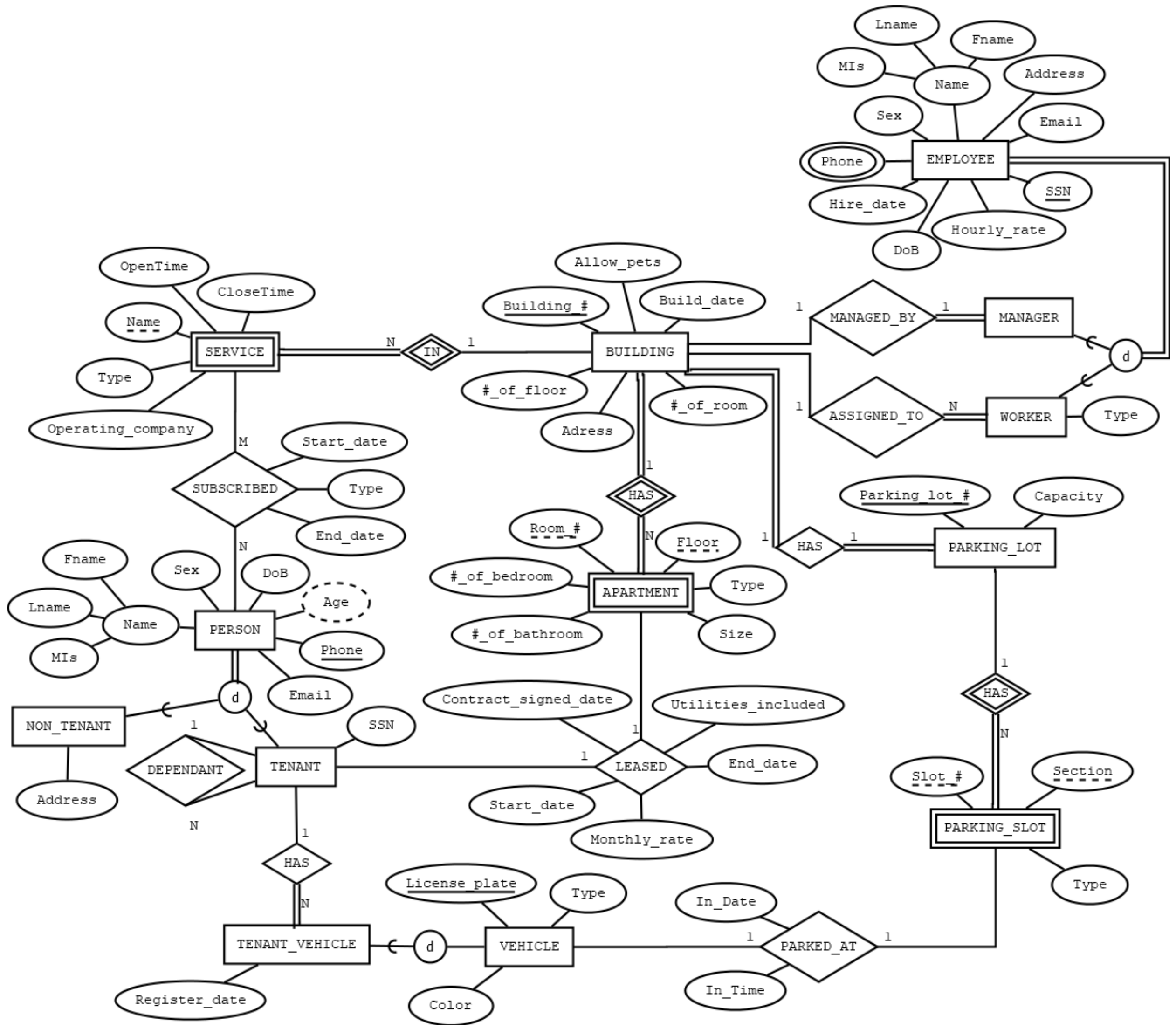
(9) A tenant must either lease the apartment or be a dependent of a tenant who is leasing the apartment. Besides keeping track of which tenant is leasing which apartment, there is the lease’s start date, end date, monthly rate, whether utilities are included in the rent, and the date the lease is signed. Noted that a tenant may only have at most 1 apartment, an apartment may only be leased by 1 tenant. Moreover, not all apartments are currently leased.

(10) For security reasons, the LAND company also wants to keep track of the tenants’ vehicles and the other vehicles that have parked in at least one of the buildings’ parking lots. For each vehicle, the company wants to keep track of the vehicle’s license plate, color, and type of vehicle (e.g., cars, mopeds, etc.,). For a tenant’s vehicle, a tenant must register it with the company, and the registered date is recorded. Noted that a tenant may have several vehicles registered and the company does not keep track of vehicles that do not require a license plate (e.g., bikes, etc.,).

(11) The LAND company also keeps track of the parking of each vehicle that is currently parked in one of the buildings’ parking lots. When a vehicle enters a parking lot, it is assigned a particular parking slot that is suitable for the vehicle. Each parking includes the date and time when the vehicle was parked, the exact slot the vehicle was in, and the license plate of the

vehicle. Noted that one slot may only hold 1 vehicle one at a time, and a vehicle can only physically be at 1 slot at a time.

II. EER Diagram



Update the EER model. The last one was the wrong version. For PERSON, Person_# should be removed, and Phone should be the key instead, and for EMPLOYEE, SSN should be the primary key, and Staff_# should be removed.

III. Preliminary Relational Schema

BUILDING(Building_#, Build_date, #_of_floor, #_of_room, Allow_pets, Address, *Parking_lot_#*)

EMPLOYEE(SSN, Fname, Lname, MIs, Sex, Email, DoB, Hourly_rate, Hire_date, Address)

EMPLOYEE_PHONE(*Employee_SSN*, Phone)

MANAGER(*Employee_SSN, Building_#*)

WORKER(*Employee_SSN*, Type, *Building_#*)

PARKING_LOT(Parking_lot_#, Capacity)

PERSON(Phone, Fname, Lname, MIs, Sex, DoB, Email)

TENANT(*Person_phone*, SSN, *Depended_tenant_phone*)

NON_TENANT(*Person_phone*, Address)

VEHICLE(License_plate, Type, Color)

TENANT_VEHICLE(*Vehicle_license_plate,* Register_date, *Tenant_phone*)

PARKING_SLOT(*Parking_lot_#*, Section, Slot_#, Type, *Vehicle_license_plate*, In_Date, In_Time)

SERVICE(*Building_#*, Name, OpenTime, CloseTime, Type, Operating_company)

APARTMENT(*Building_#*, Floor, Room_#, #_of_bedroom, #_of_bathroom, Type, Size, Lease_start_date, Lease_end_date, Lease_signed_date, Utilities_included, Monthly_rate, *Lease_tenant_phone*)

SUBSCRIBED(*Person_phone, Service_Building_#, Service_name*, Start_date, End_date, Type)

IV. Extracted Functional Dependencies

Quote	**Relation**	**Extracted Functional Dependencies**

Each building is identified by its unique building number which is assigned upon the purchase of the complex. For each building, besides the building number, the company wants to keep track of the building's build date, the number of floors and rooms in the building, the building's address, and whether the building allows pets.	BUILDING	Building_# → Build_date, #_of_floor, #_of_room, Address, Allow_pets
(By default, SSN implied people who have worked) Each employee has an SSN, name (first name, last name, and middle initials), sex, date of birth, full address, email address (which is uniquely generated on the hired date), phone numbers, the hire date, and hourly rate (in dollars).	EMPLOYEE	SSN → Fname, Lname, MIs, Sex, Email, DoB, Hourly_rate, Hire_date, Address
email address (which is uniquely generated on the hired date)	EMPLOYEE	Email → SSN

Employees’ phones are assumed to not be shared with any employee. This means a phone is unique to an employee, and there could be several phones that imply the same employee.	EMPLOYEE_PHONE	Phone → Employee_SSN
A manager must manage a particular building, and a building can only have 1 manager.	MANAGER	Employee_SSN → Building_# Building_# → Employee_SSN
A worker is assigned to a particular building. A building can have multiple workers working.	WORKER	Employee_SSN → Building_#
The LAND company also wants to keep track of the type of worker	WORKER	Employee_SSN → Type
The company also wants to keep track of the number of bathrooms and bedrooms, the apartment’s type (duplex, normal. penthouse), and the size (in squared feet) of each apartment.	APARTMENT	Building_#, Floor, Room_# → #_of_bedroom, #_of_bathroom, Type, Size

The LAND company wants to keep track of the name of the service (assumed to be unique within the same building), the daily opening and closing time of the service, the type of service, and the name of the service’s operating company.	SERVICE	Building_#, Name → OpenTime, CloseTime, Type, Operating_company
Each building has a unique parking lot.	BUILDING	Parking_lot_# → Building_# Building_# → Parking_lot_#
The parking lot shares the address with its building.	BUILDING	Parking_lot_# → Address
For each parking lot, the company wants to keep track of the parking lot number (assigned by the company and unique for each parking lot), the address, and the capacity of the parking lot.	PARKING_LOT	Parking_lot_# → Capacity, Address
For each parking slot, we also want to keep track of which type of vehicle which this particular slot is for (e.g., mopeds, bikes, cars, etc.,)	PARKING_SLOT	Parking_lot_#, Section, Slot_# → Type

Each person has a name (first name, last name, middle initials), sex, date of birth, age, phone, and email address. A person’s phone number is assumed to be unique and is used for identification upon entering the buildings and subscribing to services offered in the buildings.	PERSON	Phone → Fname, Lname, MIs, Sex, DoB, Email
email address (assumed to be unique to a person)	PERSON	Email → Phone
For each person currently subscribing to a service, the company wants to keep track of the subscription's start date, end date, and type (e.g., personal, business, etc.,).	SUBSCRIBED	Person_phone, Service_building_#, Service_name → Start_date, End_date, Type
A tenant must either lease the apartment or be a dependent of a tenant who leases the apartment.	TENANT	Person_phone → Depended_tenant_phone

For a tenant, for credit and background checks, the company keeps track of their SSN.	TENANT	Person_phone → SSN
(by default, SSNs implied people)	TENANT	SSN → Person_phone
For a non-tenant, the company further keeps track of their address.	NON_TENANT	Person_phone → Address
Besides keeping track of which tenant is leasing which apartment, there is the lease's start date, end date, monthly rate, whether utilities are included in the rent, and the date the lease is signed	APARTMENT	Lease_tenant_phone, Building_#, Floor, Room_# → Lease_start_date, Lease_end_date, Lease_signed_date, Utilities_included, Monthly_rate
Noted that a tenant may only have at most 1 apartment, an apartment may only be leased by 1 tenant	APARTMENT	Lease_tenant_phone → Building_#, Floor, Room_# Building_#, Floor, Room_# → Lease_tenant_phone
(license plate are unique to each vehicle) For each vehicle, the company wants to keep track of the vehicle's license	VEHICLE	License_plate → Type, Color

plate, color, and type of vehicle (e.g., cars, mopeds, etc.,). the company does not keep track of vehicles that do not require a license plate (e.g., bikes, etc.,).		
For a tenant’s vehicle, a tenant must register it with the company, and the registered date is recorded	TENANT_VEHICLE	Vehicle_license_plate_→ Register_date
a tenant may have several vehicles registered	TENANT_VEHICLE	Vehicle_license_plate → Tenant_phone
Each parking includes the date and time when the vehicle was parked, the exact slot the vehicle was in, and the license plate of the vehicle.	PARKING_SLOT	Vehicle_license_plate, Parking_lot_#, Section, Slot_# → Vehicle_In_Date, Vehicle_In_Time
Noted that one slot may only hold 1 vehicle one at a time, and a vehicle can only physically be at 1 slot at a time	PARKING_SLOT	Vehicle_license_plate → Parking_lot_#, Section, Slot_# Parking_lot_#, Section, Slot_# → Vehicle_license_plate

V. **3NF Check and Normalization**

1. **BUILDING(Building_#, Build_date, #_of_floor, #_of_room, Allow_pets, Address, *Parking_lot_#*)**

a. Applicable FDs

- Building_# → Build_date, #_of_floor, #_of_room, Allow_pets
- Parking_lot_# → Building_#
- Building_# → Parking_lot_#
- Parking_lot_# → Address

b. Find all candidate keys of BUILDING

- Building_#$^{+}$ = {Building_#, Build_date, #_of_floor, #_of_room, Allow_pets, Address, Parking_lot_#} = BUILDING. Therefore, Building_# is a superkey. Since Building_# is non-reducible, it is a candidate key.
- Parking_lot_#$^{+}$ = {Building_#, Build_date, #_of_floor, #_of_room, Allow_pets, Address, Parking_lot_#} = BUILDING. Therefore, Parking_lot_# is a superkey. Since Parking_lot_# is non-reducible, it is a candidate key.
- {Build_date, #_of_floor, #_of_room, Allow_pets, Address}$^{+}$ = {Build_date, #_of_floor, #_of_room, Allow_pets, Address} ≠ BUILDING. Therefore, {Build_date, #_of_floor, #_of_room, Allow_pets, Address} is not a superkey.

Therefore, the candidate keys of BUILDING are Building_# and Parking_lot_#.

c. Test if BUILDING is in 3NF

Building_#$^{+}$ = {Building_#, Build_date, #_of_floor, #_of_room, Allow_pets, Address, Parking_lot_#} = BUILDING

Parking_lot_#$^{+}$ = {Building_#, Build_date, #_of_floor, #_of_room, Allow_pets, Address, Parking_lot_#} = BUILDING

This means Building_# and Parking_lot_# are superkeys to BUILDING. Therefore, all FDs pass the 3NF test since their determinants are superkeys.

Thus, this relation is in 3NF.

2. **EMPLOYEE(SSN, Fname, Lname, MIs, Sex, Email, DoB, Hourly_rate, Hire_date, Address)**

a. Applicable FDs

- SSN → Fname, Lname, MIs, Sex, Email, DoB, Hourly_rate, Hire_date, Address (1)
- Email → SSN (2)

b. Find all candidate keys of EMPLOYEE

- SSN^+ = {Fname, Lname, MIs, Sex, Email, DoB, Hourly_rate, Hire_date, Address} = EMPLOYEE. Hence, SSN is a superkey. Since SSN is irreducible, it is a candidate key.
- $Email^+$ = {Fname, Lname, MIs, Sex, Email, DoB, Hourly_rate, Hire_date, Address} = EMPLOYEE. Hence, Email is a superkey. Since Email is irreducible, it is a candidate key.
- $\{Fname, Lname, MIs, Sex, DoB, Hourly\_rate, Hire\_date, Address\}^+$ = {Fname, Lname, MIs, Sex, DoB, Hourly_rate, Hire_date, Address}≠ EMPLOYEE. Therefore, {Fname, Lname, MIs, Sex, DoB, Hourly_rate, Hire_date, Address} is not a superkey.

Therefore, the candidate keys of EMPLOYEE are Email and SSN.

c. Test if EMPLOYEE is in 3NF

- SSN^+ = {Fname, Lname, MIs, Sex, Email, DoB, Hourly_rate, Hire_date, Address} = EMPLOYEE

- Email$^+$ = {Fname, Lname, MIs, Sex, Email, DoB, Hourly_rate, Hire_date, Address} = EMPLOYEE
- Since SSN is a superkey, FD (1) passes since its determinant is a superkey.
- Since Email is a superkey, FD (2) passes since its determinant is a superkey.

3. EMPLOYEE_PHONE(*Employee_SSN*, Phone)

a. Applicable FDs

- Phone → Employee_SSN

b. Find all candidate keys of EMPLOYEE_PHONE

- Phone$^+$ = {Phone, Employee_SSN} = EMPLOYEE_PHONE. Therefore, Phone is a superkey. Since Phone is irreducible, Phone is a candidate key.
- Employee_SSN$^+$ = {Employee_SSN} ≠ EMPLOYEE_PHONE. Therefore, Employee_SSN is not a superkey.

Therefore, the only candidate key is Phone. Since EMPLOYEE_PHONE's current key is not a candidate key, we fix the relation to only use Phone as the primary key.

EMPLOYEE_PHONE(*Employee_SSN*, Phone)

c. Test if EMPLOYEE_PHONE is in 3NF

- Since EMPLOYEE_PHONE only has 2 attributes, it is automatically in BCNF. This means that EMPLOYEE_PHONE is already in 3NF.

4. MANAGER(***Employee_SSN***, ***Building_#***)

a. Applicable FDs

- Employee_SSN → Building_#

- Building_# → Employee_SSN

b. Find all MANAGER's candidate key

- Employee_SSN$^+$ = {Employee_SSN, Building_#} = MANAGER. Therefore, Employee_SSN is a superkey. Since it is also irreducible, Employee_SSN is a candidate key.
- Building_#$^+$ = {Employee_SSN, Building_#} = MANAGER. Therefore, Building_# is a superkey. Since it is also irreducible, Building_# is a candidate key.

Therefore, the 2 candidate keys are Employee_SSN and Building_#.

c. Test if MANAGER is in 3NF

- Since MANAGER only has 2 attributes, it is automatically in BCNF. This means that MANAGER is already in 3NF.

5. WORKER(*Employee_SSN*, Type, *Building_*#)

a. Applicable FDs

- Employee_SSN → Building_#
- Employee_SSN → Type

b. Find all WORKER's candidate keys

- Employee_SSN$^+$ = {Employee_SSN, Building_#, Type} = WORKER. This means Employee_SSN is a superkey. Since Employee_SSN is irreducible, it is a candidate key.
- {Type, Building_#}$^+$ = {Type, Building_#} ≠ WORKER's. Therefore, {Type, Building_#} is not a superkey.

Therefore, the only candidate key is Employee_SSN.

c. Test if WORKER is in 3NF

Employee_SSN$^+$ = {Employee_SSN, Building_#, Type} = WORKER. This means Employee_SSN is a superkey. Therefore, both FDs pass since their determinant is a superkey. Therefore, WORKER is already in 3NF.

6. PARKING_LOT(<u>Parking_lot_#</u>, Capacity)

a. <u>Applicable FDs</u>

- Parking_lot_# → Capacity

b. <u>Find all PARKING_LOT candidate keys</u>

- Parking_lot_#$^+$ = {Parking_lot_#, Capacity} = PARKING_LOT.
- Capacity$^+$ = {Capacity} ≠ PARKING_LOT.

Therefore, the only candidate key is Parking_lot_#.

c. <u>Test if PARKING_LOT is in 3NF</u>

Parking_lot_#$^+$ = {Parking_lot_#, Capacity} = PARKING_LOT. This means the FD pass since their determinant is a superkey.

Therefore, PARKING_LOT is in 3NF.

7. PERSON(<u>Phone</u>, Fname, Lname, MIs, Sex, DoB, Email)

a. <u>Applicable FDs</u>

- Phone → Fname, Lname, MIs, Sex, DoB, Email (1)
- Email → Phone (2)

b. <u>Find all candidate keys of PERSON</u>

- Phone$^+$ = {Phone, Fname, Lname, MIs, Sex, DoB, Email} = PERSON. Therefore, Phone is a superkey. Since Phone is irreducible, it is a candidate key.

- $Email^+$ = {Phone, Fname, Lname, MIs, Sex, DoB, Email} = PERSON. Therefore, Email is a superkey. Since Email is irreducible, it is a candidate key.
- $\{Fname, Lname, MIs, Sex, DoB\}^+$ = {Fname, Lname, MIs, Sex, DoB} ≠ PERSON. Therefore, {Fname, Lname, MIs, Sex, DoB} is not a superkey.

Therefore, Phone and Email are the 2 candidate keys to PERSON.

c. Test if PERSON is in 3NF

- $Phone^+$ = {Phone, Fname, Lname, MIs, Sex, DoB, Email} = PERSON. This means the FD (1) passes since its determinant is a superkey.
- $Email^+$ = {Phone, Fname, Lname, MIs, Sex, DoB, Email} = PERSON. This means the FD (2) passes since its determinant is a superkey.

Therefore, PERSON is in 3NF.

8. TENANT(*Person_phone*, SSN, *Depended_tenant_phone*)

a. Applicable FDs

- Person_phone → Depended_tenant_phone (1)
- Person_phone → SSN (2)
- SSN → Person_phone.

b. Find all TENANT's candidate keys

- $Person\_phone^+$ = {Person_phone, SSN, Depended_tenant_phone} = TENANT. Therefore, Person_phone is a superkey. Since Person_phone is irreducible, it is a candidate key.
- SSN^+ = {Person_phone, SSN, Depended_tenant_phone} = TENANT. Therefore, SSN is a superkey. Since SSN is irreducible, it is a candidate key.

- $\text{Depended\_tenant\_phone}^+$ = {Depended_tenant_phone} ≠ TENANT. Therefore, Depended_tenant_phone is not a superkey.

Therefore, the 2 candidate keys are Person_phone and SSN.

c. Test if TENANT is in 3NF

- Person_phone^+ = {Person_phone, SSN, Depended_tenant_phone} = TENANT. Therefore, both FDs (1) and (2) pass since their determinant is a superkey.
- SSN^+ = {Person_phone, SSN, Depended_tenant_phone} = TENANT. Therefore, FD (3) passes since their determinant is a superkey.

Therefore, TENANT is in 3NF.

9. NON_TENANT(*Person_phone*, Address)

a. Applicable FDs

- Person_phone → Address

b. Find all candidate keys for NON_TENANT

- Person_phone^+ = {Person_phone, Address} = NON_TENANT. Therefore, Person_phone is a superkey. Since Person_phone is irreducible, Person_phone is a candidate key.
- Address^+ = {Address} ≠ NON_TENANT. Therefore, Address is not a superkey.

Therefore, the only candidate key is Person_phone.

c. Test if NON_TENANT is in 3NF

- Person_phone^+ = {Person_phone, Address} = NON_TENANT. Therefore, the FD passes since its determinant is a superkey.

Therefore, NON_TENANT is in 3NF.

10. VEHICLE(License_plate, Type, Color)

a. Applicable FDs

- License_plate → Type, Color

b. Find all candidate keys for VEHICLE

- License_plate is the candidate key, because License_plate$^+$ = {License_plate, Type, Color} = VEHICLE and it is irreducible.

c. Test if VEHICLE is in 3NF

- Since License_plate is the relation primary key, License_plate is a superkey. This means the FD passed because its determinant is a superkey.

Therefore, VEHICLE is in 3NF.

11. TENANT_VEHICLE(*Vehicle_license_plate*, Register_date, *Tenant_phone*)

a. Applicable FDs

- Vehicle_license_plate → Register_date
- Vehicle_license_plate → Tenant_phone

b. Find all candidate keys for TENANT_VEHICLE

- Vehicle_license_plate$^+$ = {Vehicle_license_plate, Register_date, Tenant_phone} = TENANT_VEHICLE. Thus, Vehicle_license_plate is the candidate key.

c. Test if TENANT_VEHICLE is in 3NF

- Vehicle_license_plate is TENANT_VEHICLE's superkey, as it is VEHICLE's candidate key. Therefore, both FDs passed the conditions requiring TENANT_VEHICLE to be in 3NF.

12. PARKING_SLOT(*Parking_lot_#*, Section, Slot_#, Type, *Vehicle_license_plate*, Vehicle_In_Date, Vehicle_In_Time)

a. Applicable FDs

- Parking_lot_#, Section, Slot_# → Type
- Vehicle_license_plate, Parking_lot_#, Section, Slot_# → Vehicle_In_Date, Vehicle_In_Time
- Vehicle_license_plate → Parking_lot_#, Section, Slot_#
- Parking_lot_#, Section, Slot_# → Vehicle_license_plate

b. Find all candidate keys for PARKING_SLOT

- Vehicle_license_plate$^+$ = {Vehicle_license_plate, Parking_lot_#, Slot_#, Section, Type, Vehicle_In_Date, Vehicle_In_Time} = PARKING_SLOT. Therefore, Vehicle_license_plate is the entity's candidate key.
- {Parking_lot_#, Section, Slot_#}$^+$ = {Vehicle_license_plate, Parking_lot_#, Slot_#, Section, Type, Vehicle_In_Date, Vehicle_In_Time} = PARKING_SLOT. Therefore, Parking_lot_#, Section, Slot_# is the entity's candidate key.

c. Test if PARKING_SLOT is in 3NF

- Parking_lot_#, Section, Slot_# → Type

  Parking_lot_#, Section, Slot_# → Vehicle_license_plate

  As {Parking_lot_#, Section, Slot_#} is a candidate key, it is a superkey. Thus, the two FDs satisfy the conditions requiring PARKING_SLOT to be in 3NF.
- Vehicle_license_plate, Parking_lot_#, Section, Slot_# → Vehicle_In_Date, Vehicle_In_Time

{Vehicle_license_plate, Parking_lot_#, Section, Slot_#}$^+$ = {Vehicle_license_plate, Parking_lot_#, Slot_#, Section, Type, Vehicle_In_Date, Vehicle_In_Time} = PARKING_SLOT. Thus, {Vehicle_license_plate, Parking_lot_#, Section, Slot_#} is a candidate key. Therefore, the FD satisfies the conditions requiring PARKING_SLOT to be in 3NF.

- Vehicle_license_plate → Parking_lot_#, Section, Slot_#

  As Vehicle_license_plate is a candidate key, it is a superkey. Therefore, the FD satisfies the conditions requiring PARKING_SLOT to be in 3NF.

13. SERVICE(***Building_#***, **Name,** OpenTime, CloseTime, Type, Operating_company)

a. Applicable FDs

- Building_#, Name → OpenTime, CloseTime, Type, Operating_company

b. Find all candidate keys for SERVICE

- {Building_#, Name}$^+$ = {Building_#, Name, OpenTime, CloseTime, Type, Operating_company} = SERVICE. Building_# is not a superkey, as {Building_#}$^+$ = {Building_#}. Name is not a superkey, as {Name}$^+$ = {Name}. As a result, {Building_#, Name} is a candidate key.

c. Test if SERVICE is in 3NF

- As {Building_#, Name} is the SERVICE's candidate key, the FD satisfies the conditions requiring SERVICE to be in 3NF.

14. APARTMENT(*Building_#*, Floor, Room_#, #_of_bedroom, #_of_bathroom, Type, Size, Lease_start_date, Lease_end_date, Lease_signed_date, Utilities_included, Monthly_rate, *Lease_tenant_phone*)

a. Applicable FDs

- Building_#, Floor, Room_# → #_of_bedroom, #_of_bathroom, Type, Size (1)
- Lease_tenant_phone, Building_#, Floor, Room_# → Lease_start_date, Lease_end_date, Lease_signed_date, Utilities_included, Monthly_rate (2)
- Lease_tenant_phone → Building_#, Floor, Room_# (3)
- Building_#, Floor, Room_# → Lease_tenant_phone (4)

b. Find all candidate keys for APARTMENT

- {Building_#, Floor, Room_#}$^{+}$ = {Building_#, Floor, Room_#, #_of_bedroom, #_of_bathroom, Type, Size, Lease_start_date, Lease_end_date, Lease_signed_date, Utilities_included, Monthly_rate, Lease_tenant_phone} = APARTMENT. Therefore, {Building_#, Floor, Room_#} is a superkey. We want to check if any subsets of it a superkey.
    - {Building_#, Floor}$^{+}$ = {Building_#, Floor} ≠ APARTMENT
    - {Building_#, Room_#}$^{+}$ = {Building_#, Room} ≠ APARTMENT
    - {Floor, Room_#}$^{+}$ = {Floor, Room_#} ≠ APARTMENT
- Therefore, {Building_#, Floor, Room_#} is a candidate key.
- Lease_tenant_phone^{+} = {Lease_tenant_phone, Building_#, Floor, Room_#, …} = APARTMENT. Hence, Lease_tenant_phone is a superkey, and since it is irreducible, it is a candidate key.

- {#_of_bedroom, #_of_bathroom, Type, Size, Lease_start_date, Lease_end_date, Lease_signed_date, Utilities_included, Monthly_rate, Building_#, Floor} = {#_of_bedroom, #_of_bathroom, Type, Size, Lease_start_date, Lease_end_date, Lease_signed_date, Utilities_included, Monthly_rate, Building_#, Floor} ≠ APARTMENT. Therefore, it is not a superkey.
- {#_of_bedroom, #_of_bathroom, Type, Size, Lease_start_date, Lease_end_date, Lease_signed_date, Utilities_included, Monthly_rate, Floor, Room_#} = {#_of_bedroom, #_of_bathroom, Type, Size, Lease_start_date, Lease_end_date, Lease_signed_date, Utilities_included, Monthly_rate, Floor, Room_#} ≠ APARTMENT. Therefore, it is not a superkey.
- {#_of_bedroom, #_of_bathroom, Type, Size, Lease_start_date, Lease_end_date, Lease_signed_date, Utilities_included, Monthly_rate, Building_#, Room_#} = {#_of_bedroom, #_of_bathroom, Type, Size, Lease_start_date, Lease_end_date, Lease_signed_date, Utilities_included, Monthly_rate, Building_#, Room_#} ≠ APARTMENT. Therefore, it is not a superkey.

  Therefore, the 2 candidate keys for APARTMENT are Lease_tenant_phone and {Building_#, Floor, Room_#}.

c. Test if APARTMENT is in 3NF

- We have that {Building_#, Floor, Room_#} is a superkey. Hence, FD (1) and (4) pass the 3NF test.
- We have that {Lease_tenant_phone} is a superkey. Hence, FD (3) passes the test.

  {Lease_tenant_phone, Building_#, Floor, Room_#}$^+$ = {Lease_tenant_phone, …}.

Therefore, {Lease_tenant_phone, Building_#, Floor, Room_#} is a superkey. Therefore, FD (2) passes the 3NF test.

- Therefore, APARTMENT is in 3NF.

15. SUBSCRIBED(*<u>Person_phone, Service_building_#, Service_name</u>*, Start_date, End_date, Type)

a. <u>Applicable FDs</u>

- Person_phone, Service_building_#, Service_name → Start_date, End_date, Type

b. <u>Find all candidate keys for SUBSCRIBED</u>

- {Person_phone, Service_building_#, Service_name}$^+$ = {Person_phone, Service_building_#, Service_name, Start_date, End_date, Type} = SUBSCRIBED. Therefore, {Person_phone, Service_building_#, Service_name} is a superkey.
  - {Person_phone, Service_building_#}$^+$ = {Person_phone, Service_building_#} ≠ SUBSCRIBED.
  - {Service_building_#, Service_name}$^+$ = {Service_building_#, Service_name} ≠ SUBSCRIBED.
  - {Person_phone, Service_name}$^+$ = {Person_phone, Service_name} ≠ SUBSCRIBED.
- Therefore, {Person_phone, Service_building_#, Service_name} is a candidate key.
- {Start_date, End_date, Type, Person_phone, Service_building_#} = {Start_date, End_date, Type, Person_phone, Service_building_#} ≠ SUBSCRIBED. Therefore, {Start_date, End_date, Type, Person_phone, Service_building_#} is not a superkey.

- {Start_date, End_date, Type, Service_building_#, Service_name} = {Start_date, End_date, Type, Service_building_#, Service_name} ≠ SUBSCRIBED. Therefore, {Start_date, End_date, Type, Service_building_#, Service_name} is not a superkey.
- {Start_date, End_date, Type, Person_phone, Service_name} = {Start_date, End_date, Type, Person_phone, Service_name} ≠ SUBSCRIBED. Therefore, {Start_date, End_date, Type, Person_phone, Service_namee} is not a superkey.

Therefore, the only candidate key is {Person_phone, Service_building_#, Service_name}.

c. Test if SUBSCRIBED is in 3NF

Since {Person_phone, Service_building_#, Service_name} is a superkey, the FD passes since its determinant is a superkey.

Therefore, SUBSCRIBED is in 3NF.

Note: On a side note, all of these relations are also in BCNF because we never need to use the prime attribute condition in any of the relations.

VI. Results

We have that the preliminary relational schema is already in 3NF. Therefore, we do not have to do any normalization. Hence, the final schema is as follows:

BUILDING(Building_#, Build_date, #_of_floor, #_of_room, Allow_pets, Address, *Parking_lot_#*)

EMPLOYEE(SSN, Fname, Lname, MIs, Sex, Email, DoB, Hourly_rate, Hire_date, Address)

EMPLOYEE_PHONE(*Employee_SSN*, Phone) (since Phone itself made the key)

MANAGER(*Employee_SSN*, *Building_#*)

WORKER(*Employee_SSN*, Type, *Building_#*)

PARKING_LOT(Parking_lot_#, Capacity)

PERSON(Phone, Fname, Lname, MIs, Sex, DoB, Email)

TENANT(*Person_phone*, SSN, *Depended_tenant_phone*)

NON_TENANT(*Person_phone*, Address)

VEHICLE(License_plate, Type, Color)

TENANT_VEHICLE(*Vehicle_license_plate*, Register_date, *Tenant_phone*)

PARKING_SLOT(*Parking_lot_#*, Section, Slot_#, Type, *Vehicle_license_plate*, Vehicle_In_Date, Vehicle_In_Time)

SERVICE(*Building_#*, Name, OpenTime, CloseTime, Type, Operating_company)

APARTMENT(*Building_#*, Floor, Room_#, #_of_bedroom, #_of_bathroom, Type, Size, Lease_start_date, Lease_end_date, Lease_signed_date, Utilities_included, Monthly_rate, *Lease_tenant_phone*)

SUBSCRIBED(*Person_phone, Service_Building_#, Service_name*, Start_date, End_date, Type)

VII. Adjustments

For the APARTMENT table, if it is not leased, values for fields holding information about the lease will be NULL. Hence, systematically, this table has lots of NULL. Therefore, to reduce NULL in the database, we decided to decompose the APARTMENT table as follows:

APARTMENT(*Building_#*, Floor, Room_#, #_of_bedroom, #_of_bathroom, Type, Size)

RENTED_APARTMENT(*Building_#, Floor, Room_#,* Lease_start_date, Lease_end_date, Lease_signed_date, Utilities_included, Monthly_rate, *Lease_tenant_phone*)