JIMMA UNIVERSITY
JIMMA INSTITUTE OF TECHNOLOGY
FACULTY OF COMPUTING AND INFORMATICS
DEPARTMENT OF SOFTWARE ENGINEERING

ADVANCED DATABASE SYSTEMS (SEng 3041)

Name of student	ID No
1. AYANA GEMEDA	RU0318/15
2. ABENEZER BEKELE	RU0118/15
3. DULA GUDETA	RU0647/15
4. JITU ALEMAYEHU	RU1150/15
5. USMAEL TAJU	RU1061/14
6. FULMI TESFAYE	RU0871/15
7. WADA ABERA	RU2095/15
8. NYAKAN PETER	RU2303/15
9. WORKAYEHU MELESE	RU2119/15

January, 2025
Jimma, Ethiopia

1. ER Diagram

1.1 ER Diagram Description

ER diagram is used to provide an in-depth description of the entities, relationships, and attributes in the ER diagram for the home rentals system. Emphasize the importance of understanding the data structure and relationships to ensure the system's functionality and efficiency.

Entities

It represented by Rectangle. It is an object or a concept that exists independently and can be distinctly identified. In a graph, entities are typically represented as nodes or vertices, and they can be anything from physical objects like people and products to abstract concepts like events or categories. Dedicate a subsection to each entity, discussing their roles and responsibilities in more detail:

1. **Homeowner**: Discuss the responsibilities of a homeowner, such as listing homes for rent, reviewing tenant applications, and maintaining properties.

2. **Tenan**t: Explain the role of a tenant, including searching for homes, submitting applications, paying rent, and reporting issues.

3. **Police Station:** Highlight the importance of police stations in ensuring the safety and security of the rental properties and their role in addressing crime-related concerns.

4. **Home:** Discuss the significance of the Home entity in storing and managing rental property information.

Relationships

It is a diamond shape. A relationship describes how two or more entities are connected or interact with each other. In a graph, relationships are represented as edges or links between nodes. These relationships can be directional or bidirectional and can also have attributes to describe the nature or strength of the connection. Dedicate a subsection to each relationship, exploring their purposes and how they facilitate interactions between entities:

1. **Lease:** Explain the significance of the lease agreement, including its role in binding homeowners and tenants to specific terms and conditions.

2. **Notify:** Describe how the Notify relationship fosters communication and cooperation between homeowners, tenants, and police stations to maintain safety and address security concerns.

3. **Owns:** Elaborate on the importance of establishing clear ownership of rental properties for legal and management purposes.

Attributes

An attribute is an oval shape property or characteristic of an entity that provides more information about it. Attributes are usually represented as labels or data associated with the nodes in a graph. Discuss each entity's attributes in greater detail, focusing on their data types, constraints, and purpose within the system:

- **Home**: Talk about how home-related attributes, such as rent_amount and number_of_rooms, impact tenants' rental decisions.

- **Homeowner:** Discuss how homeowner attributes, such as contact_info, facilitate communication and management.

- **Tenant**: Explain the significance of tenant attributes like start_date and previous_address for rental applications and background checks.

- **Police Station:** Highlight the importance of accurate station_id and contact_info attributes for efficient communication and response in emergencies.

In the following sections, we will delve into the specifics of how attributes and entities are represented and related within a graph. Understanding these relationships is crucial for accurately interpreting the graph and extracting meaningful insights from the data it represents. We will explore the definitions of entities and attributes, the nature of their connections, and how these connections can be leveraged to gain a deeper comprehension of the underlying data structure. This sets the stage for a detailed discussion on the relationship between attributes and entities within the graph.

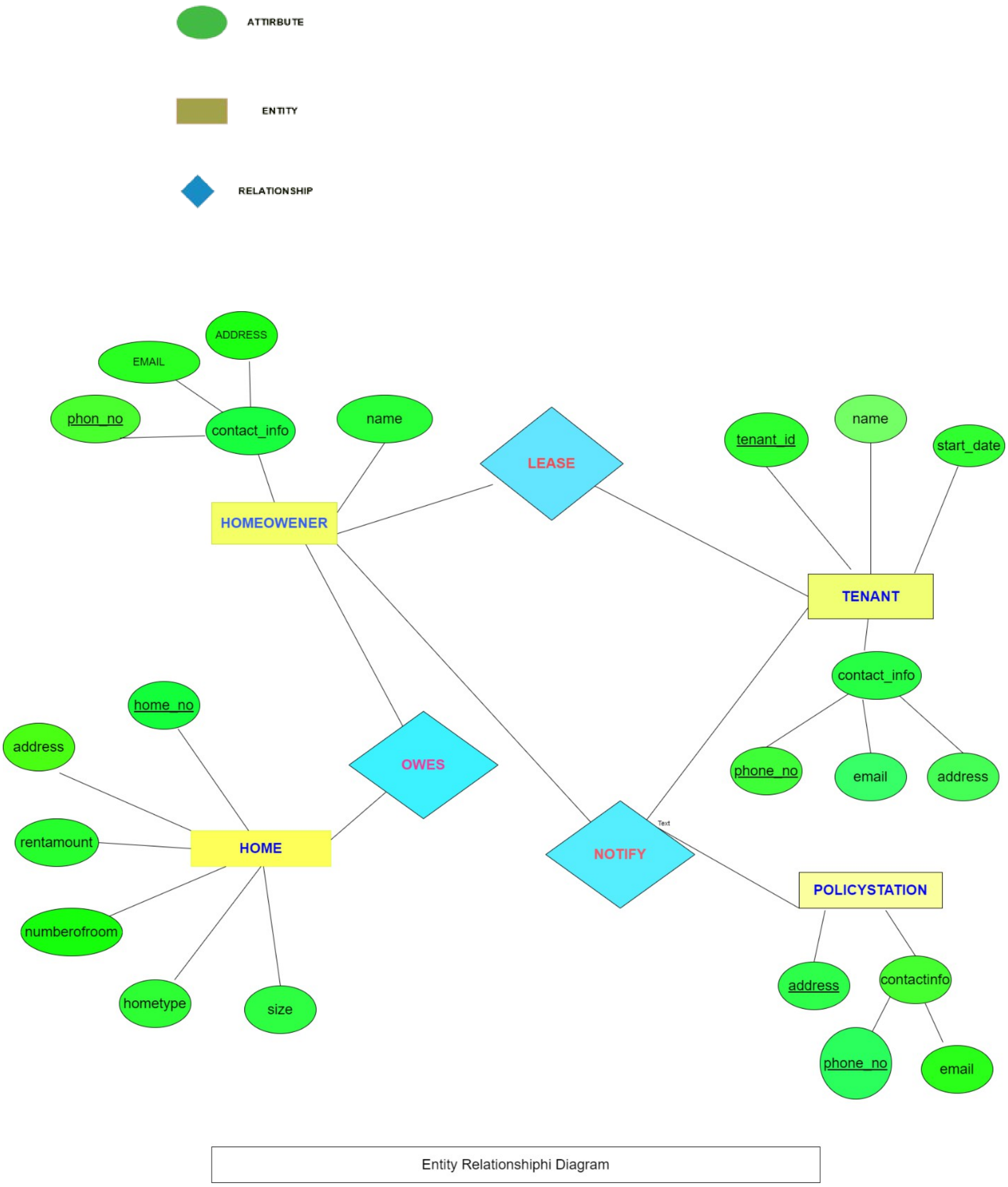


Figure 1:ER Diagram

1.2 Use Case

The key actors in the system are the HomeOwner, PoliceStation, and Tenant. Here is a detailed textual explanation of the diagram:

Actors:

1. **HomeOwner** (represented by a yellow circle)
2. **PoliceStation** (represented by a blue circle)
3. **Tenant** (represented by a green circle)

Use Cases:

1. **List Available Homes** - HomeOwner can list the homes available for rent.
2. **Update Home Details** - HomeOwner can update the details of a home.
3. **Delete Home** - HomeOwner can delete a home from the list.
4. **Add Home** - HomeOwner can add a new home to the list.
5. **Notify The Police** - HomeOwner can notify the police station.
6. **Approve Rent Request** - HomeOwner can approve a rent request made by a Tenant.
7. **Logout** - A general action available to all actors to log out of the system.
8. **Login** - A general action available to all actors to log into the system.
9. **View Rental History** - Tenant can view their rental history.
10. **Contact Home Owner** - Tenant can contact the HomeOwner.
11. **Update Budget** - Tenant can update their budget preferences.
12. **Request To Rent** - Tenant can request to rent a home.
13. **View Home Details** - Tenant can view details of available homes.
14. **See Notification** - PoliceStation can see notifications related to home rentals.
15. **View Tenant Details** - PoliceStation can view details of tenants.

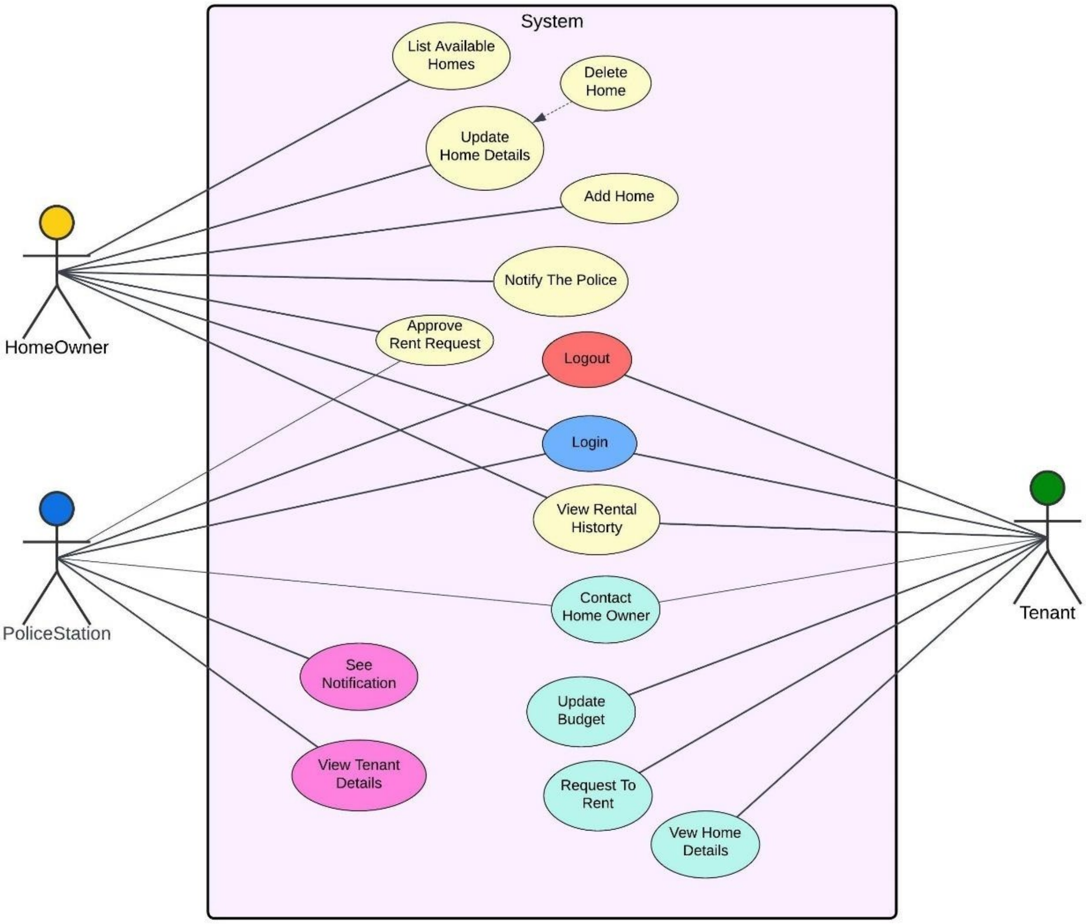


Figure 2:Use Case Diagram

1.3 Class Diagram

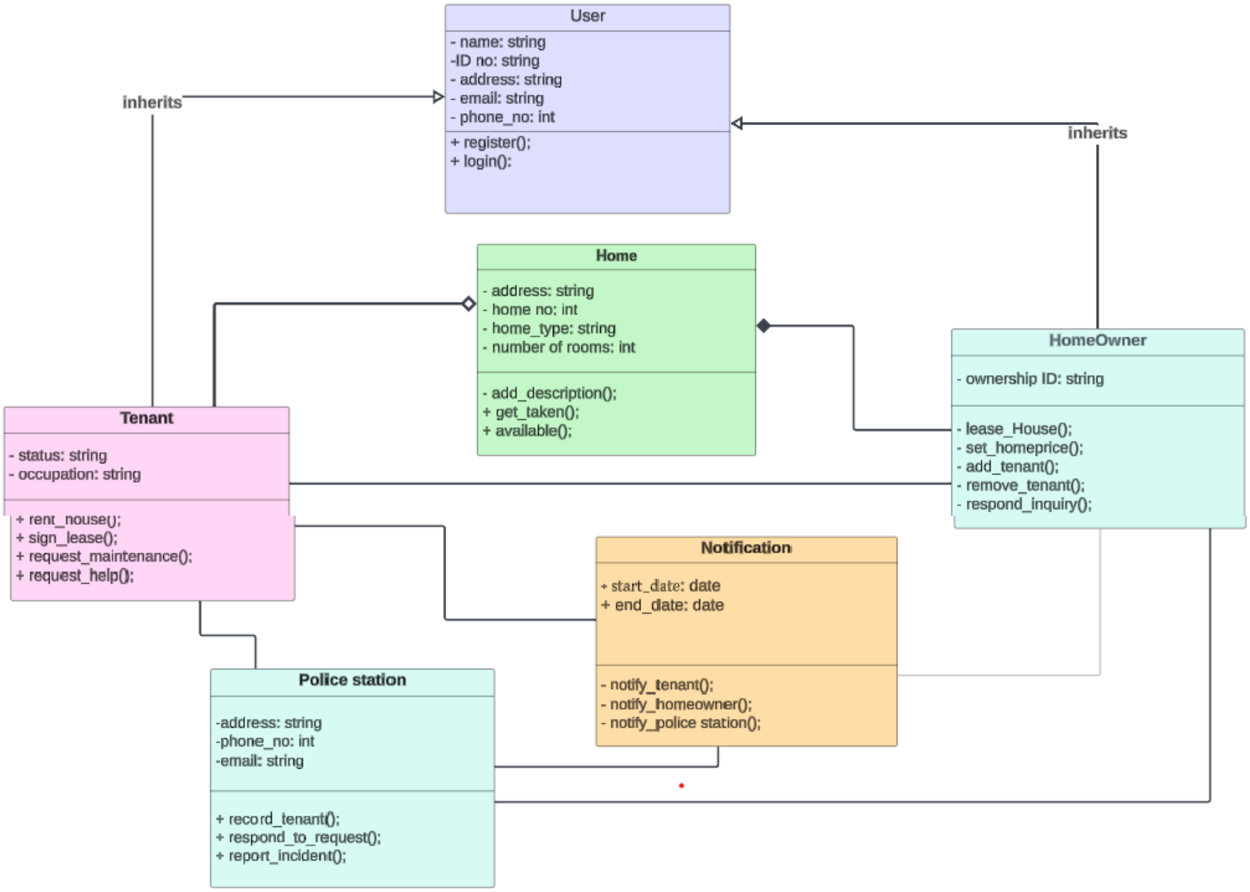


Figure 3:Class Diagram

1.4 Component Diagram

This diagram is a component diagram for a Community Home Rental System. It depicts the main components of the system and their interactions, focusing on data access, security, and notifications. Below is a detailed description of each component and their relationships, suitable for inclusion in a project paper.

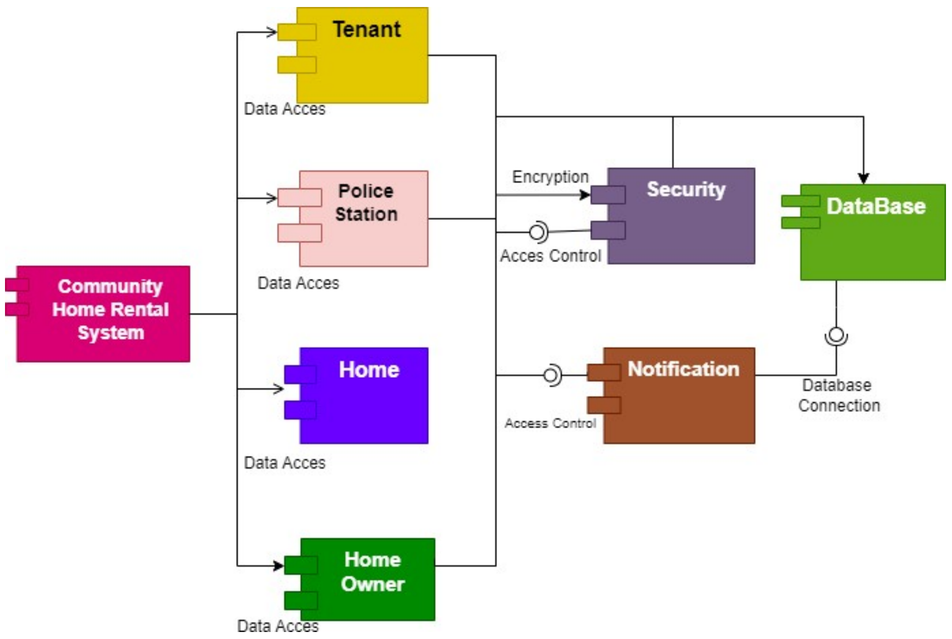


Figure 4:Component Diagram

1.4.1 Components and Descriptions:

1. **Community Home Rental System**
   - **Description**: The main system that interacts with various components to manage home rentals in a community.
   - **Interactions**: Connects with Tenant, Police Station, Home, and Home Owner components for data access.
2. **Tenant**
   - **Description**: Represents the tenants who use the system to find and rent homes.
   - **Interactions**: Accesses data related to available homes and their details.
3. **Police Station**
   - **Description**: Represents the police station which can view notifications and tenant details.
   - **Interactions**: Accesses data related to notifications and tenant information.
4. **Home**
   - **Description**: Represents the home entity within the system, including home details and availability.

5. **Home Owner**
    - **Description**: Represents homeowners who list and manage homes available for rent.
    - **Interactions**: Accesses data to update, add, or delete home information.
6. **Security**
    - **Description**: Ensures data encryption and access control within the system.
    - **Interactions**: Manages secure data access for all components, connecting to the database securely.
7. **Notification**
    - **Description**: Manages notifications sent to tenants and police stations.
    - **Interactions**: Controls access to notifications and ensures proper delivery.
8. **Database**
    - **Description**: Stores all data related to tenants, homes, homeowners, and notifications.
    - **Interactions**: Connected to the Security component for encrypted data access and storage.

1.4.2 Relationships and Interactions:

9. **Data Access**:
    - **Tenant, Police Station, Home, Home Owner**: All these components access and interact with data within the system.
    - **Home Owner**: Manages and updates home data within the system.
    - **Police Station**: Accesses tenant details and receives notifications.
    - **Tenant**: Views home details and rental history.
10. **Security**:
    - Ensures encryption and access control for all data transactions.
    - Interacts with the Database to provide secure data storage and access.
11. **Notification**:
    - Manages access control for sending and receiving notifications.
    - Interacts with the Database to retrieve and store notification data.
12. **Database Connection**:
    - The Database component is central to the system, providing data storage for all other components.

- Connected securely through the Security component to ensure data integrity and confidentiality.

1.5 NORMALIZATION

We are going to normalize each table based on insertion, updating and deletion for each entity that wants normalizate. We have to normalize each table by using normalization rules to optimize the table.

The table's that need normalization are

- Home owner
- Tenant
- Home
- Police stations

1.5.1 First Normal Form

- The first normal form requires that the data in a table be stored in a tabular format, with no repeating groups or nested data.
- Each cell in the table should contain a single, atomic value.

1.5.2 Second Normal Form

- The second normal form builds on the first normal form and requires that all non-key attributes be fully dependent on the primary key.
- This means that there should be no partial dependencies,

1.5.3 Third Normal Form

- The third normal form builds on the second normal form and requires that all non-key attributes be independent of each other and depend only on the primary key.
- This means that there should be no transitive dependencies,

1.5.4 Normalization of The HomeOwner Table

HomeOwner ('HomOw_ID', 'hom_FullName', 'homOw_Address', 'homOw_PhoneNumber', 'Homow_Email',homow_contact info)

First Normal Form (1NF)**:**

- HomeOwner (`HomOw_ID`, `homow_FullName’, ‘homOw_Address`, homOw_PhoneNumber’, ‘Homow_Email`,homow_contact info)
- The table should contain atomic values .so we have to create separate table for address
- I.e Adress(homOw_id, kebele, house_no, city)
- Primary Key: `HomOw_ID` (unique identifier for each home owner)
- There are no repeating groups, and each attribute contains a single value.

Second Normal Form (2NF):

- HomeOwner(homOw_id,homow_FullName`, homow_Address’, ‘homw_PhoneNumber’, homw_`Email`,homw_contact info)
- There are no partial dependencies in this table
- all non-key attributes are fully dependent on the primary key, `HomeOw_ID`.

Third Normal Form (3NF):

- If there are any non-key attributes that determine other non-key attributes, you would need to create a separate table to eliminate the transitive dependency.

All tables are:

- Homow_Address (homOw_ID, kebele, house_no, city,Street,and `Zip)
- Homow_Phone (homow_id, phon1, phon2)
- -There are no transitive dependencies,

1.5.5 Normalization Of Home Table

Home(`HomeID`, `Address`, `no of rooms`, `Rent`, `HomeType`, `Size`)

First Normal Form (1NF):

- There are no repeating groups, and each attribute contains a single value.
- Home(`HomeID`, `Address`, `no of rooms`, `Rent`, `HomeType`, `Size`)
- Primary Key: `HomeID` (unique identifier for each home)

Second Normal Form (2NF):

- There are no partial dependencies.
- I.e Home (`Address`, `no of rooms `, `Rent`, `HomeType`, `Size`)

Third Normal Form (3NF):

- There are no transitive dependencies,
- Any such dependencies are eliminated by creating a separate table.
- For example, if the `HomeType` and `Size` attributes could be broken down into more detailed characteristics, you would create separate tables (e.g., `HomeTypeDetails`, `SizeDetails`) and establish relationships between the `Home`, `HomeTypeDetails`, and `SiteDetails` tables.

1.5.6 Normalization for The Tenant Table

Tenant (<u>Ten_Id</u>,FullName, Address, PhoneNumber, Email, StartDate, EndDate)

First Normal Form (1NF):

- There are no repeating groups, and each attribute contains a single value.
- I.e Tenant (Ten<u>_ID</u>, FullName, Address, PhoneNumber, Email, StartDate, EndDate)
- TenantID (unique identifier for each tena***nt)***

Second Normal Form (2NF):

- There are no partial dependencies in this table.
- all non-key attributesare fully dependent on the primary key, TenantID
- I.e Tenant (<u>Ten_id</u>,FullName, Address, PhoneNumber, Email, StartDate, EndDate)

Third Normal Form (3NF):

- Eliminating transitive dependency or eliminate if non-key attribute determines another non-key attribute

All tables are,

- I.e -TenantAddress (<u>ten_ID</u>,ten_Kebele,ten_HouseNo,ten_City

1.5.7 Normalization of The Police Stations Table

PoliceStation(`Pol_ID`, `pol_Address`, `pol_contact info`)

First Normal Form (1NF)

- There are no repeating groups, and each attribute contains a single value
- PoliceStation(`Pol_ID`, `pol_Address`, `pol_contact info`)
- Primary Key: `PoliceStationID` (unique identifier for each police station)

Second N5ormal Form (2NF):

all non-key attributes are fully dependent on the primary key, Pol_ID.

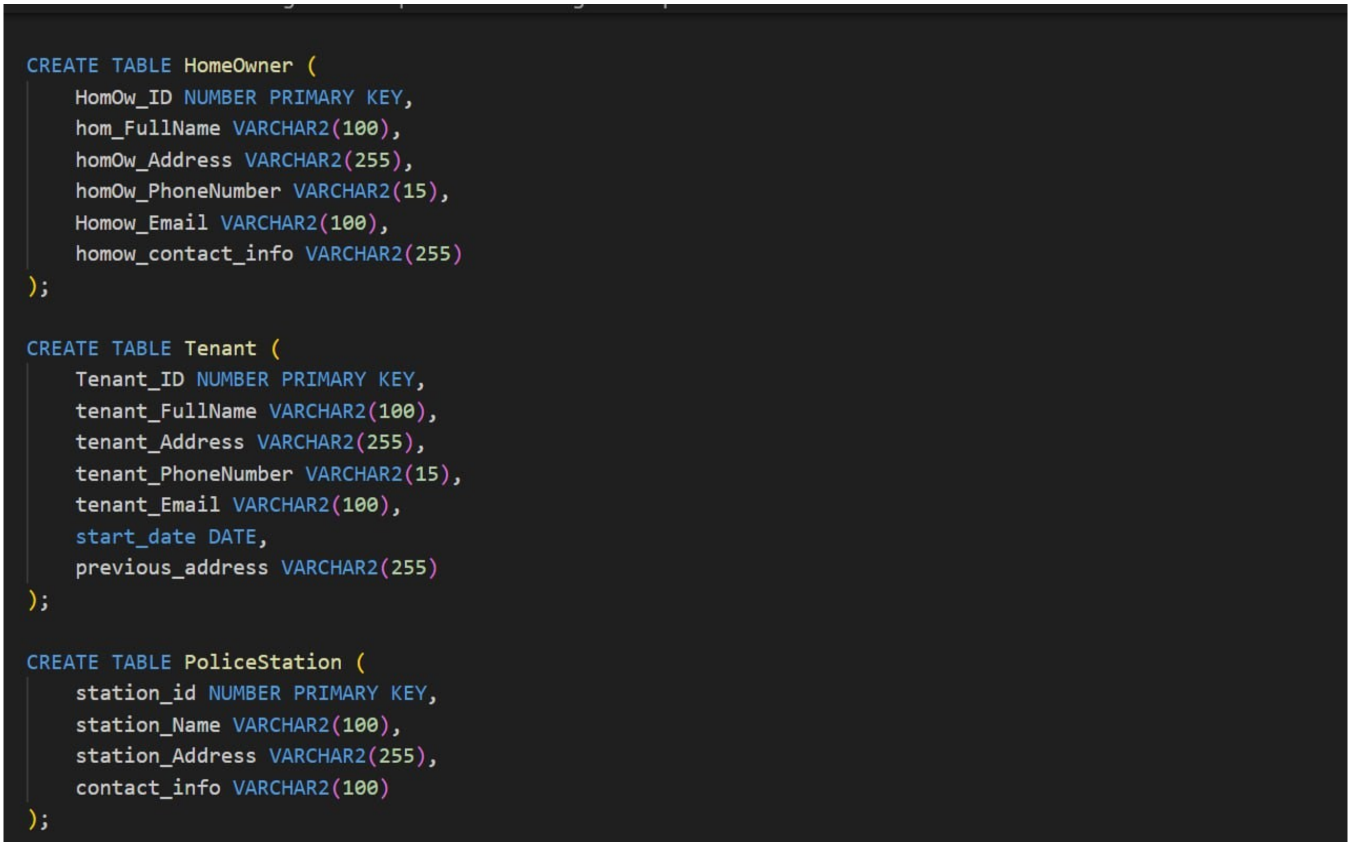
- -There are no partial dependencies,
- I.e PoliceStation(pol _id,`Address,Pol_contactinfo)

Third Normal Form (3NF):

- -If there are any non-key attributes that determine other non-key attributes, you would need to create a separate table to eliminate the transitive dependency.

All tables are;

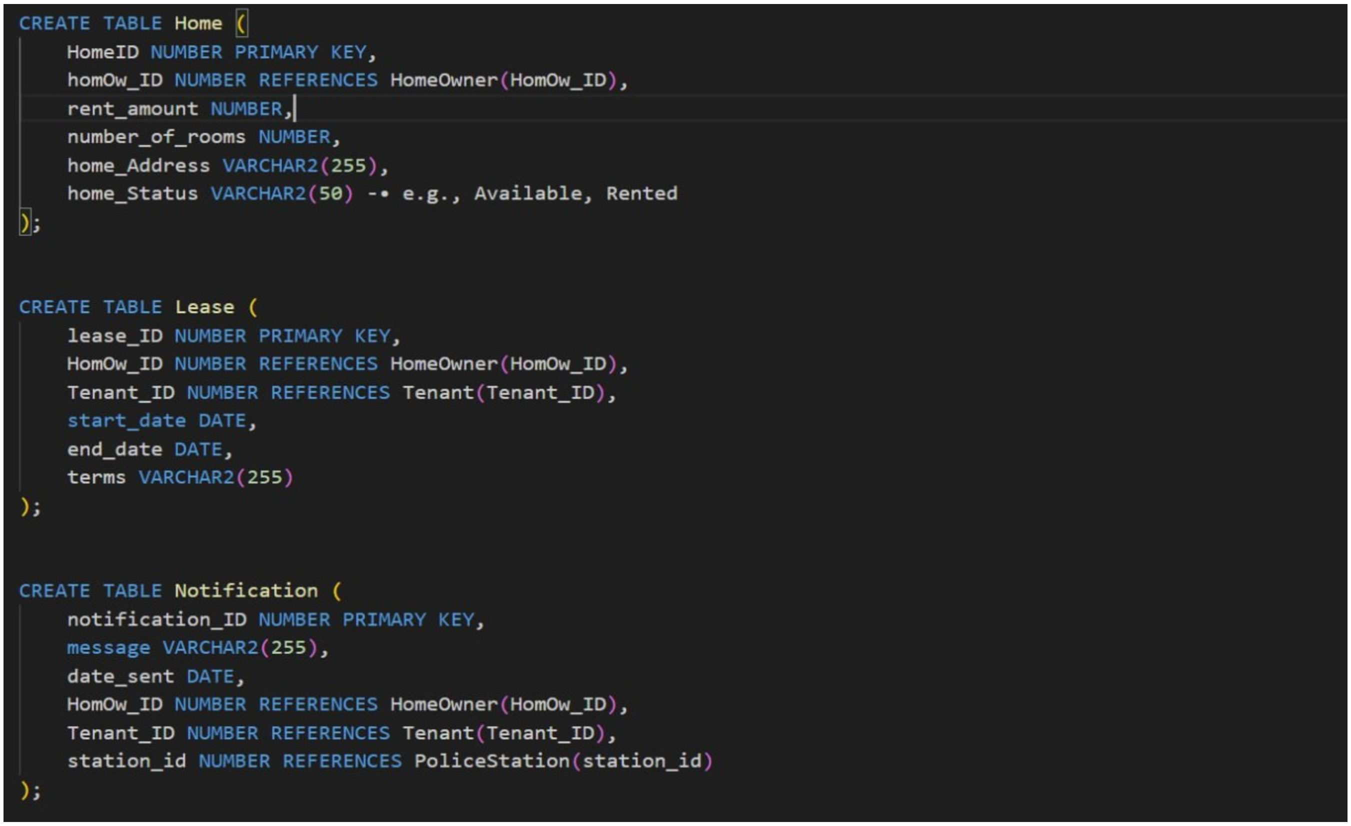
- pol_Address(pol_id, kebele, street_, city)
- Pol_contactinfo(pol_Emai ,pol_PhoneNumber



```sql
CREATE TABLE HomeOwner (
    HomOw_ID NUMBER PRIMARY KEY,
    hom_FullName VARCHAR2(100),
    homOw_Address VARCHAR2(255),
    homOw_PhoneNumber VARCHAR2(15),
    Homow_Email VARCHAR2(100),
    homow_contact_info VARCHAR2(255)
);

CREATE TABLE Tenant (
    Tenant_ID NUMBER PRIMARY KEY,
    tenant_FullName VARCHAR2(100),
    tenant_Address VARCHAR2(255),
    tenant_PhoneNumber VARCHAR2(15),
    tenant_Email VARCHAR2(100),
    start_date DATE,
    previous_address VARCHAR2(255)
);

CREATE TABLE PoliceStation (
    station_id NUMBER PRIMARY KEY,
    station_Name VARCHAR2(100),
    station_Address VARCHAR2(255),
    contact_info VARCHAR2(100)
);
```



```sql
CREATE TABLE Home (
    HomeID NUMBER PRIMARY KEY,
    homOw_ID NUMBER REFERENCES HomeOwner(HomOw_ID),
    rent_amount NUMBER,
    number_of_rooms NUMBER,
    home_Address VARCHAR2(255),
    home_Status VARCHAR2(50) -• e.g., Available, Rented
);


CREATE TABLE Lease (
    lease_ID NUMBER PRIMARY KEY,
    HomOw_ID NUMBER REFERENCES HomeOwner(HomOw_ID),
    Tenant_ID NUMBER REFERENCES Tenant(Tenant_ID),
    start_date DATE,
    end_date DATE,
    terms VARCHAR2(255)
);


CREATE TABLE Notification (
    notification_ID NUMBER PRIMARY KEY,
    message VARCHAR2(255),
    date_sent DATE,
    HomOw_ID NUMBER REFERENCES HomeOwner(HomOw_ID),
    Tenant_ID NUMBER REFERENCES Tenant(Tenant_ID),
    station_id NUMBER REFERENCES PoliceStation(station_id)
);
```