**Title: Property Rental System in UK**

The scope of a Property Rental System is to facilitate the management of rental properties, tenants, landlords, and rental agreements. The system helps to smooth the process of property rental, ensuring efficient communication between property agents, landlords, and tenants.

**The key components and functionalities of the system include:**

**Property Management:** The system should allow users to add, view, update, and delete rental properties. Each property's details should be stored and easily accessible.

**Landlord Information:** The system should store and manage landlord details, including their names, contact and addresses. It should also keep a record of properties.

**Tenant Information:** The system should get tenant details, such as names, contact and addresses. It should allow tenants to specify their rental preferences.

**Rental Agreements:** The system should record rental agreements between tenants and landlords.

**Property Agents and Agencies**: The system should store information about property agents. It should track properties managed by each agent.

**Maintenance Requests:** The system should provide a mechanism for tenants to submit maintenance requests. It should record the description, status, and the resolution.

**Rental Payment Processing:** The system should support rental payment processing and tracking. It should allow property agents and agencies to record rent payments.

**Reporting and Analytics:** The system should provide reporting capabilities to generate statements and reports.

**Data Security:** The system should implement appropriate data security measures to protect sensitive information.

**Tasks**

**Task 1 - Conceptual Design in ER**

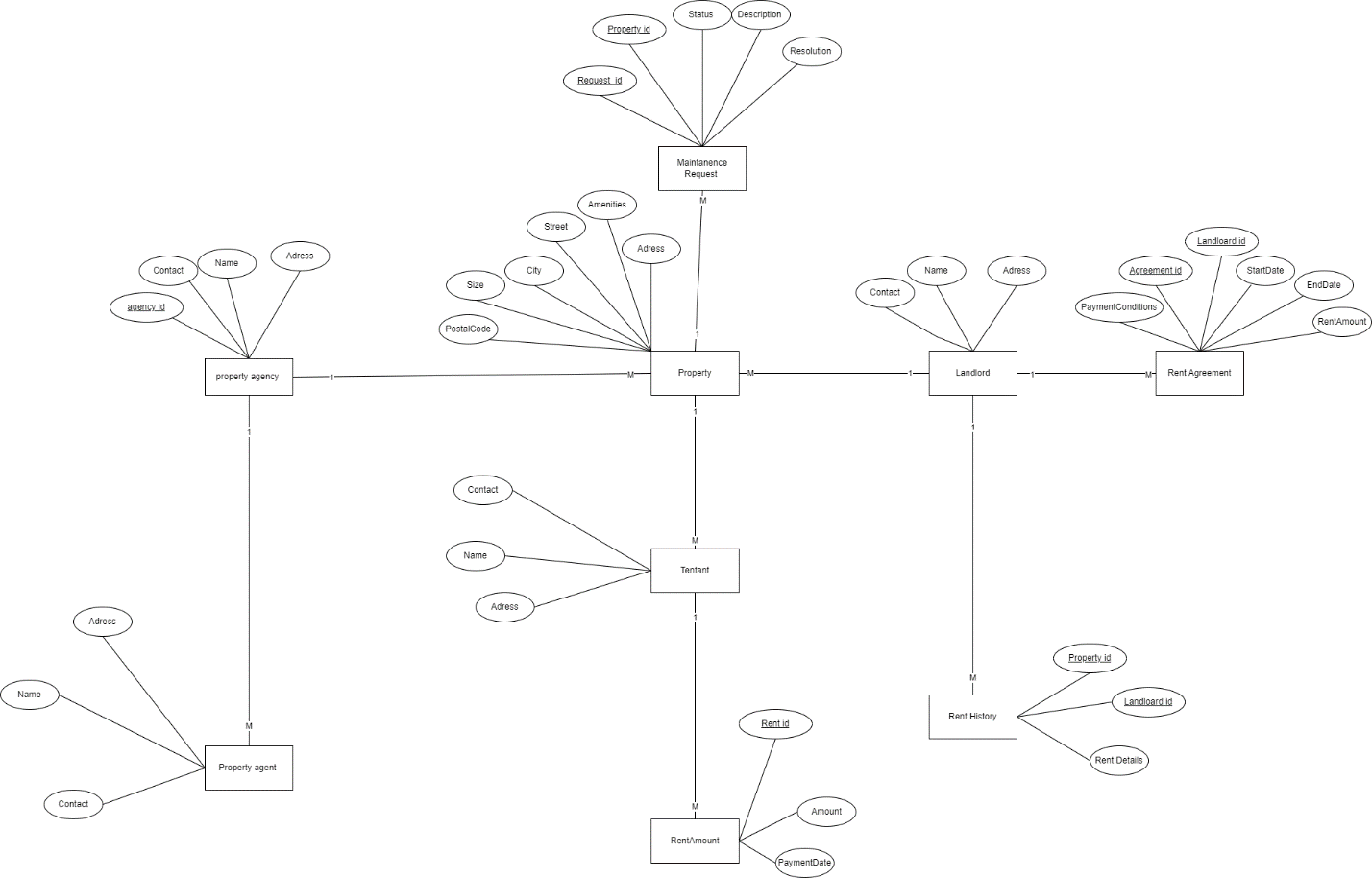


Figure 1.1

**Task 2 - Logical Design in Relational**

|  |  |
| --- | --- |
| Elements in ER Diagram | which are represented by Elements in Relational Schema |
| entity Property | relation Property  Primary Key: PropertyID  Relationship: Landlord  Attributes: PostalCode,size,City,Street,Ameneties,Adress |
| entity Landlord | relation Landlord  Primary Key: LandlordIdID  Relationship: Property  Attributes: Name,Contact,Adress |
| entity Tenant | relation Tenant  Primary Key: TenantID  Relationship: Property  Attributes: Name,Contact,Adress |
| entity PropertAgent | relation PropertyAgent  Primary Key: AgentID  Relationship: PropertyAgency  Attributes: Name,Contact,Adress |
| entity PropertyAgency | relation PropertyAgency  Primary Key: AgencyID  Relationship: Property  Attributes: Name,Contact,Adress |
| entity RentAmount | relation RentAmount  Primary Key: RentID  Relationship: Tenant  Attributes: Amount,PaymentDate |
| entity RentHistory | relation RentHistory  Primary Key: HistoryID  Relationship: Landlord  Attributes: RentDetails  Foreign Keys:PropertyID, LandlordID |
| entity MaintenanceRequest | relation MaintenanceRequest  Primary Key: RequestID  Relationship: Property  Attributes: Status, Description, Resolution  Foreign Keys:PropertyID |

Description:

Entities: Each entity in the diagram is represented as a table in the relational schema.

Primary Key: The primary key attribute for each table is shown. It uniquely identifies each record in the table.

Attributes: The attributes of each entity are listed in the "Attributes". There are no multivalued attributes.

Foreign Key: If an entity has a relationship with another entity, the foreign key attribute is included to reference the primary key of the related entity.

Relationship: The relationships between entities are represented in the "Relationship".

Relational Diagram of Property Rental System :

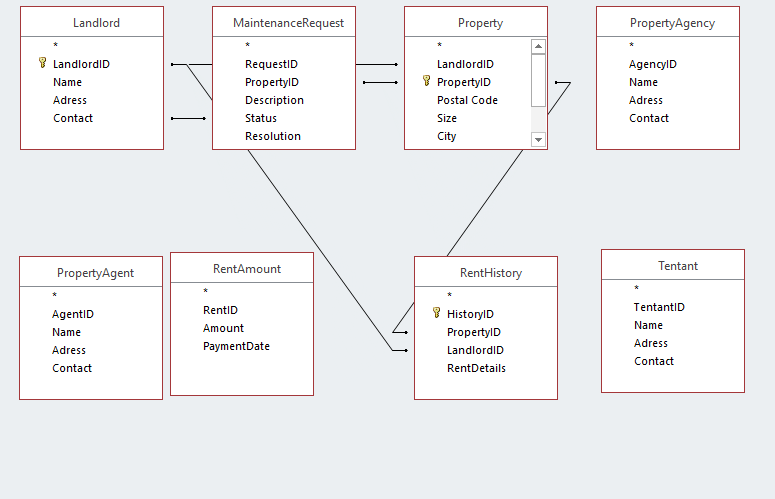


Figure: Relational Diagram of the System

Relational schema using the notation is shown below:

**Property(**

PropertyID **(PK),**

street,

city,

size,

postalcode,

amenities

**)**

**Landlord(**

landlordID (PK),

name,

contact,

Adress

**)**

**Tenant(**

tenantID **(PK),**

name,

contact,

Adress

**)**

**PropertyAgent(**

**agentID (PK),**

name,

contact,

Adress

**)**

**PropertyAgency(**

**agencyID(PK),**

name,

contact,

Adress

**)**

**RentAmount(**

rentID (PK),

Amount,

PaymentDate

**)**

**RentHistory(**

**historyID (PK),**

PropertyID **(FK),**

landlordID (FK),

**)**

**MaintenanceRequest(**

**requestID (PK),**

PropertyID **(FK),**

description,

status,

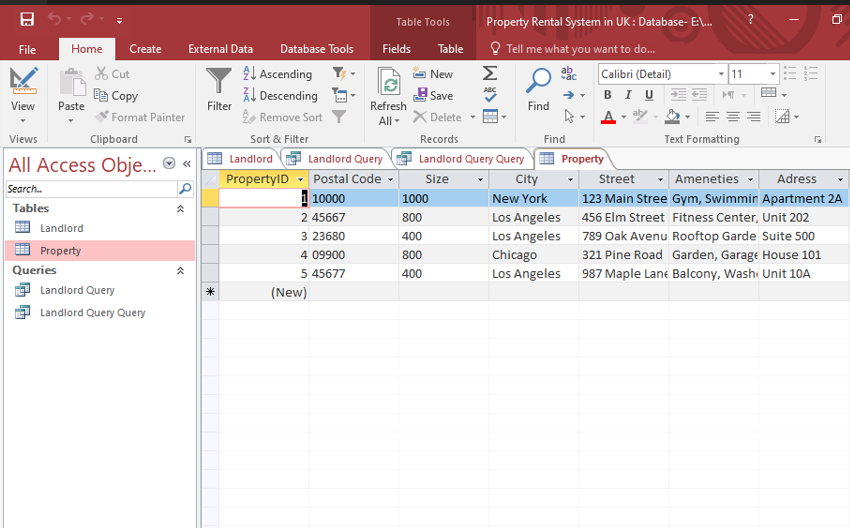
resolution

**)**

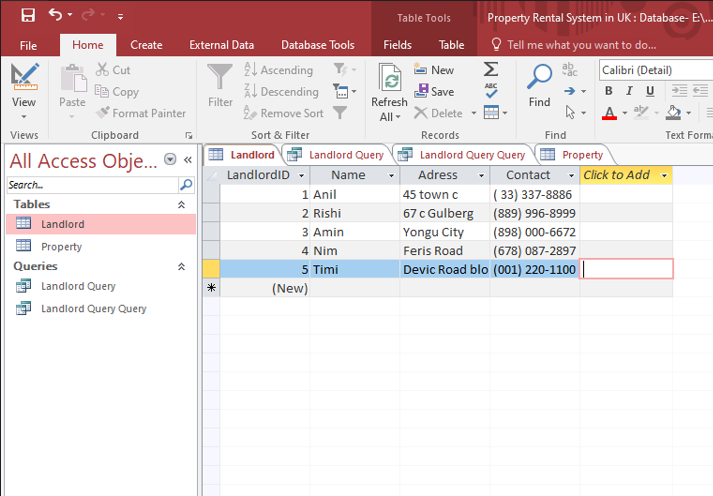
**Task 3 - Implement the Database**

**Task 4 – Populate the Tables with Data**

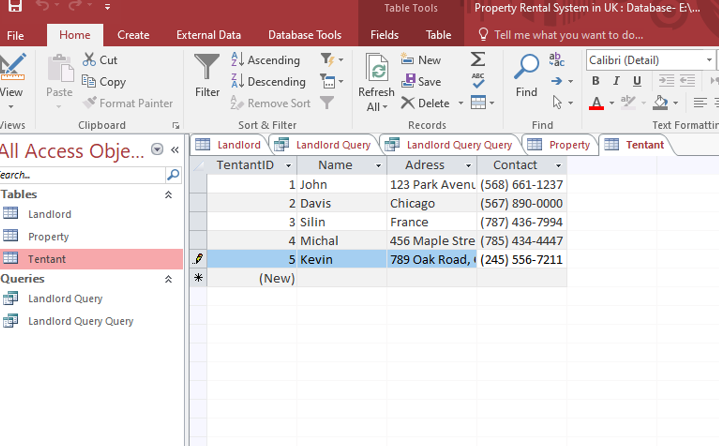
**Property:**



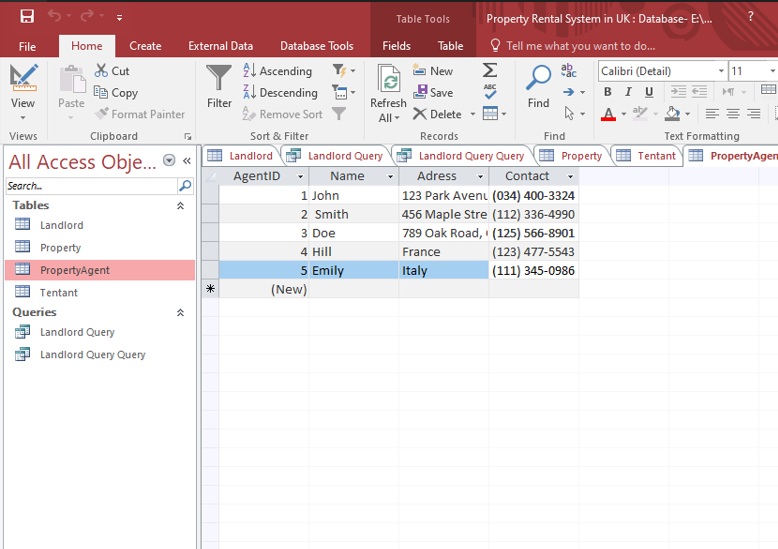
**Landlord:**



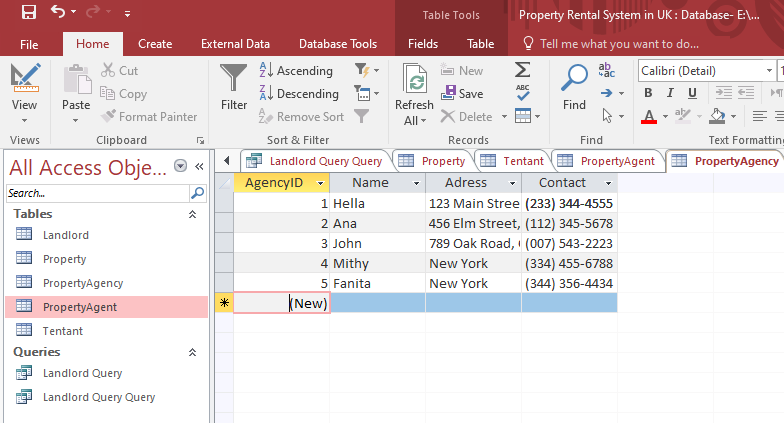
**Tentant:**

****

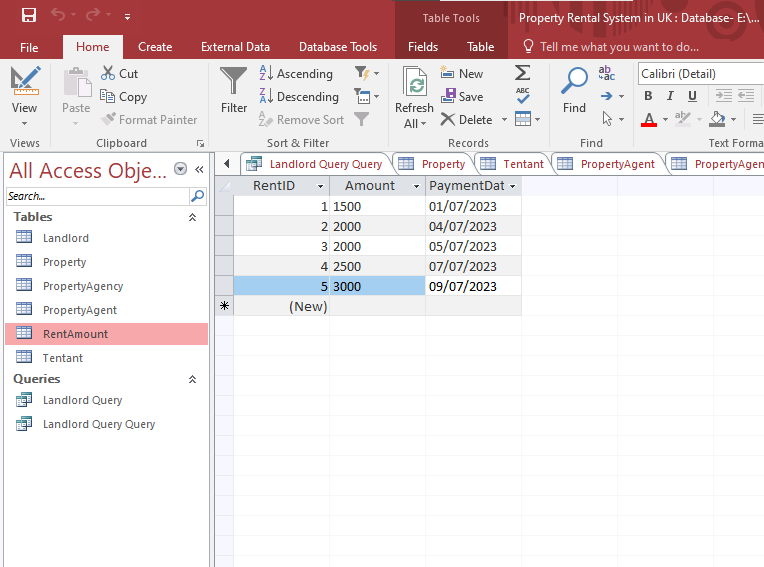
**PropertyAgent:**

****

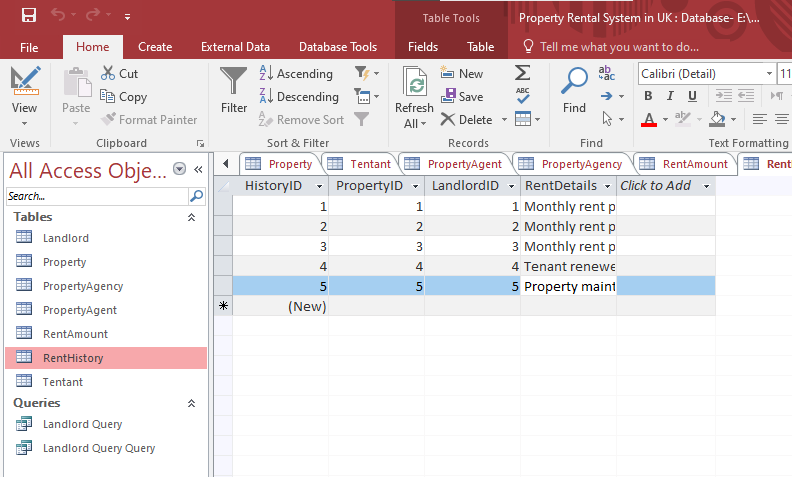
**PropertyAgency:**

****

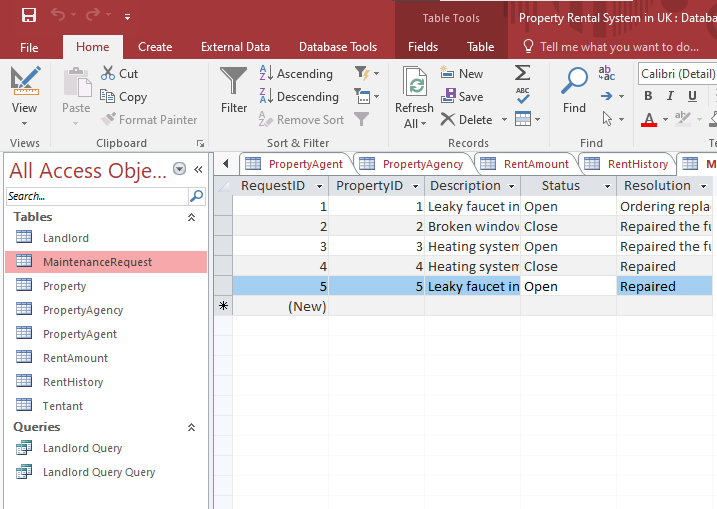
**RentAmount:**

****

**RentHistory:**

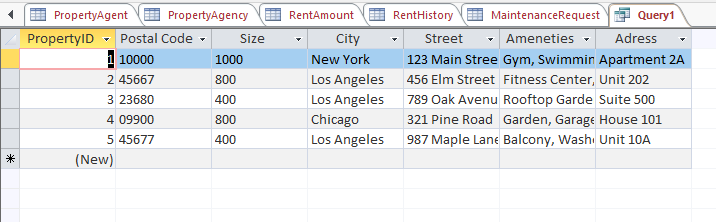
****

**MaintenanceRequest**

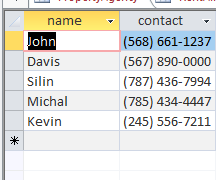
****

**Task 5 - Query the Database**

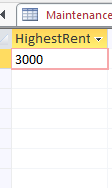
1:SELECT \* FROM Property;



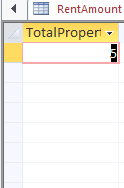
2:SELECT name, contact FROM Tenant;



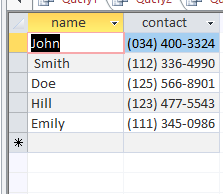
3: SELECT MAX(Amount) AS HighestRent FROM RentAmount;



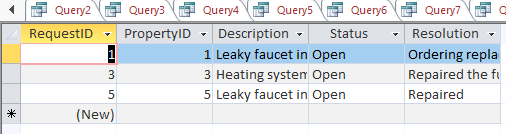
4: SELECT COUNT(\*) AS TotalProperties FROM Property;



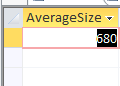
5:SELECT name, contact FROM PropertyAgent;



6: SELECT \* FROM MaintenanceRequest WHERE status = 'Closed';



7: SELECT AVG(Size) AS AverageSize FROM Property;

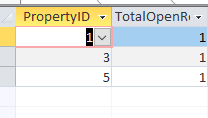


8: SELECT PropertyID, COUNT(\*) AS TotalOpenRequests

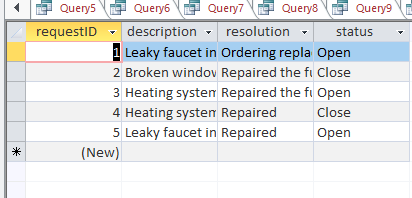
FROM MaintenanceRequest

WHERE status = 'Open'

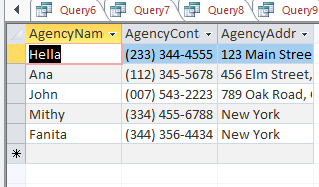
GROUP BY PropertyID;



9: SELECT requestID, description, resolution, status FROM MaintenanceRequest;



10: SELECT name AS AgencyName, contact AS AgencyContact, Adress AS AgencyAddress FROM PropertyAgency;



**Task 6 - Database Application**

Develop and implement an application that will allow the database users to access and retrieve data from the database. The application should have a 'user friendly' graphical interface. The application should allow the users to perform the following:

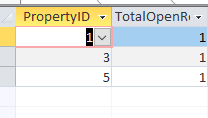
1. Run the last 3 queries in task 5;

1: SELECT PropertyID, COUNT(\*) AS TotalOpenRequests

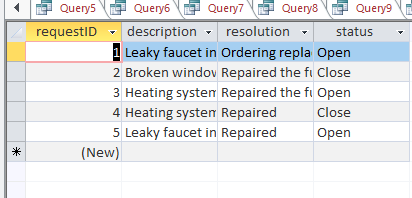
FROM MaintenanceRequest

WHERE status = 'Open'

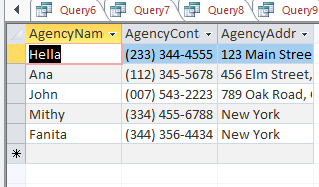
GROUP BY PropertyID;



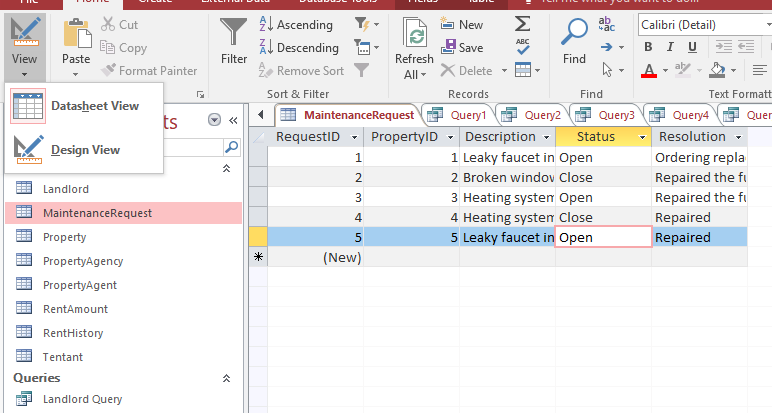
2: SELECT requestID, description, resolution, status FROM MaintenanceRequest;

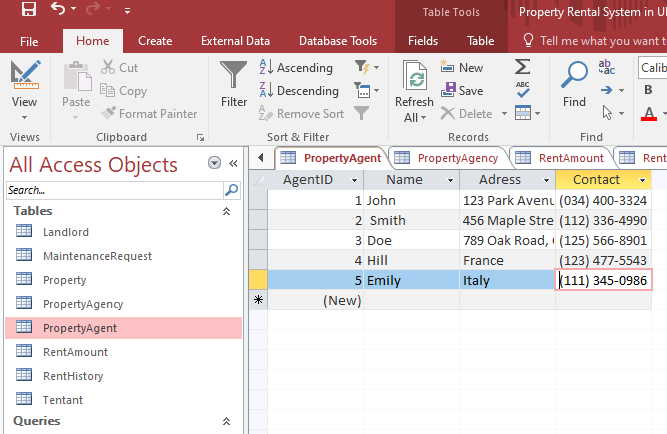


3: SELECT name AS AgencyName, contact AS AgencyContact, Adress AS AgencyAddress FROM PropertyAgency;



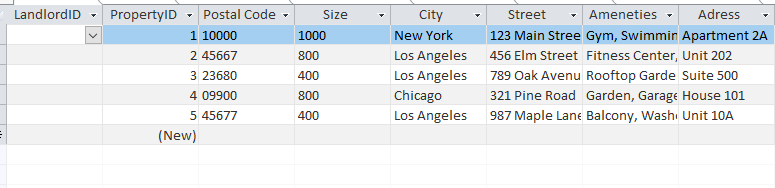
1. View data in 2 tables of your choice in datasheet view;



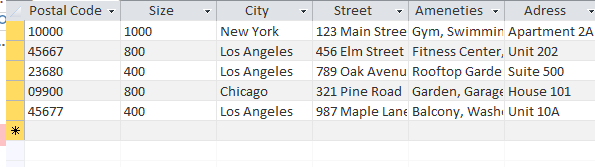


1. Update data: using multi-table forms users should be able to add, update and delete data from at least two tables

1: Updated Property Table (added fields):



2: Updated Property Table (deleted fields):



**Task 7 - Critical Evaluation**

The value of coursework related to designing and implementing a relational database lies in its practical application. Understanding the principles and techniques of database design is important for creating efficient, scalable, and maintainable systems for managing data.

**Key points of value in this coursework include:**

**Data Organization**: Learning how to structure data in a logical and organized manner is crucial for efficient data retrieval and manipulation.

**Normalization**: Normalization techniques help minimize data redundancy and improve data integrity. It helps to design databases that are less prone to errors.

**Data Integrity and Constraints**: Implementing data integrity through primary keys, foreign keys, and constraints ensures that the data remains accurate and consistent.

**Querying and Optimization**: SQL queries writing and their optimization can significantly impact the performance of database operations. Efficiently retrieving data using SQL enhances the overall system performance.

**Indexing and Performance Tuning**: Understanding indexing and performance tuning techniques is critical for large-scale databases. Properly designed indexes can speed up data retrieval.

**Data Security**: Database design involves considerations for data security. This helps protect sensitive information and ensures compliance with data protection regulations.

**Real-World Applications**: The coursework provides valuable insights into applying database concepts to real-world scenarios, such as building applications for managing rental properties, customer data, financial transactions, etc.

**My Contribution:**

During the database design phase of the coursework, I actively participated in identifying the entities, attributes, and relationships required for the rental property management system. I collaborated with other team members to create an initial ERD. In the implementation phase, my primary responsibility was to set up the database schema using Microsoft Access. I designed and created the tables.