Housing Management System Project

Sayali Shelke

Supriya Thorat

Atharva Kulkarni

Surabhi Chavan

San Diego State University

MIS686-01:Enterprise Database Management

Dr. Morteza Safaei-Pour

13 December 2024

Abstract

The **Housing Management System** is a solution designed to efficiently manage housing societies, properties, tenants, and financial transactions. It automates tasks such as rent payments, maintenance tracking, and tenant management, improving operational efficiency.

The system features a relational database with entities like **Society**, **Property**, **Tenant**, **Rent Payment**, and **Maintenance request** interconnected through relationships. SQL queries (DDL, DML), views, triggers, and stored procedures manage and manipulate the data.

A **dashboard** provides analytical insights into rent trends, maintenance requests, and property occupancy, aiding in data-driven decision-making. The system is deployed on a cloud platform with role-based user access for different permissions. The project enhances the efficiency and effectiveness of property management tasks.

Introduction

Purpose: Explain that the project involves creating a comprehensive database for managing housing societies, properties, tenants, rent payments, and maintenance requests. This system aims to automate and streamline housing management tasks.

Why this project?: Highlight how this project ties into real-world applications, improving efficiency in property management for agencies handling multiple societies and tenants. Create a new paragraph to begin the block-quote. Double-space each line of text, as you have done with headings, section labels, and paragraphs of paraphrased text. To correctly format a block-quote, indent each line of the text to one-half inch. Remember to always cite your source.

Objective

The **Housing Management System** project aims to develop a database-driven solution to efficiently manage housing societies, properties, tenants, and related processes. The key objectives are:

1. **Automate Property Management**: Track property details, tenant information, and payments.
2. **Improve Efficiency**: Streamline rent payments, maintenance requests, and tenant management.
3. **Provide Insights**: Generate analytical reports on occupancy rates, revenue, and maintenance trends.
4. **Support Scalability**: Design a system that can scale as the agency expands.
5. **Role-Based Access**: Implement secure user access control for data integrity.

   Cloud Deployment: Deploy the system on AWS for reliability and accessibility.

Applications

The **Housing Management System** project aims to develop a database-driven solution to efficiently manage housing societies, properties, tenants, and related processes. The key objectives are: This property management system can be applied in various real-world scenarios to streamline operations and improve efficiency. Below are the key applications for this project:

---

1. Property Management Agencies

- **Purpose**: Simplify and automate the management of rental properties across multiple societies.
- **Key Features**:
  - Keep track of available and rented properties.

  - Maintain tenant records, including rent payment history.
  - Generate rental income reports and manage late fee penalties.
  - Log and monitor maintenance requests to ensure timely resolutions.

---

2. Residential Societies and Housing Associations

- **Purpose**: Manage residential properties and provide services to residents.
- **Key Features**:
  - Track properties within the society (e.g., apartments, villas).
  - Manage tenant and owner information.
  - Handle maintenance requests for facilities like elevators, water supply, etc.
  - Provide a platform for rent payment processing.

---

3. Commercial Real Estate Firms

- **Purpose**: Manage commercial properties like office spaces, retail stores, and warehouses.
- **Key Features**:
  - Maintain a database of commercial properties with size, rent, and status.
  - Schedule and record rent payments from businesses.
  - Track tenant details and agreements.
  - Log maintenance requests for commercial infrastructure.

---

4. Freelance Property Managers

- **Purpose**: Assist independent property managers in tracking and managing a small number of properties.
- **Key Features**:

  - Maintain a centralized database of properties and tenants.
  - Automate payment reminders and receipts.
  - Monitor maintenance issues and communicate with service providers.
  - Generate reports for property owners.

---

5. Facility Maintenance Teams

- **Purpose**: Manage and prioritize maintenance tasks for properties.
- **Key Features**:
  - Receive maintenance requests directly from tenants.
  - Assign tasks to team members and track progress.
  - Ensure timely resolution of maintenance issues with real-time updates.
  - Generate reports to analyze common maintenance problems.

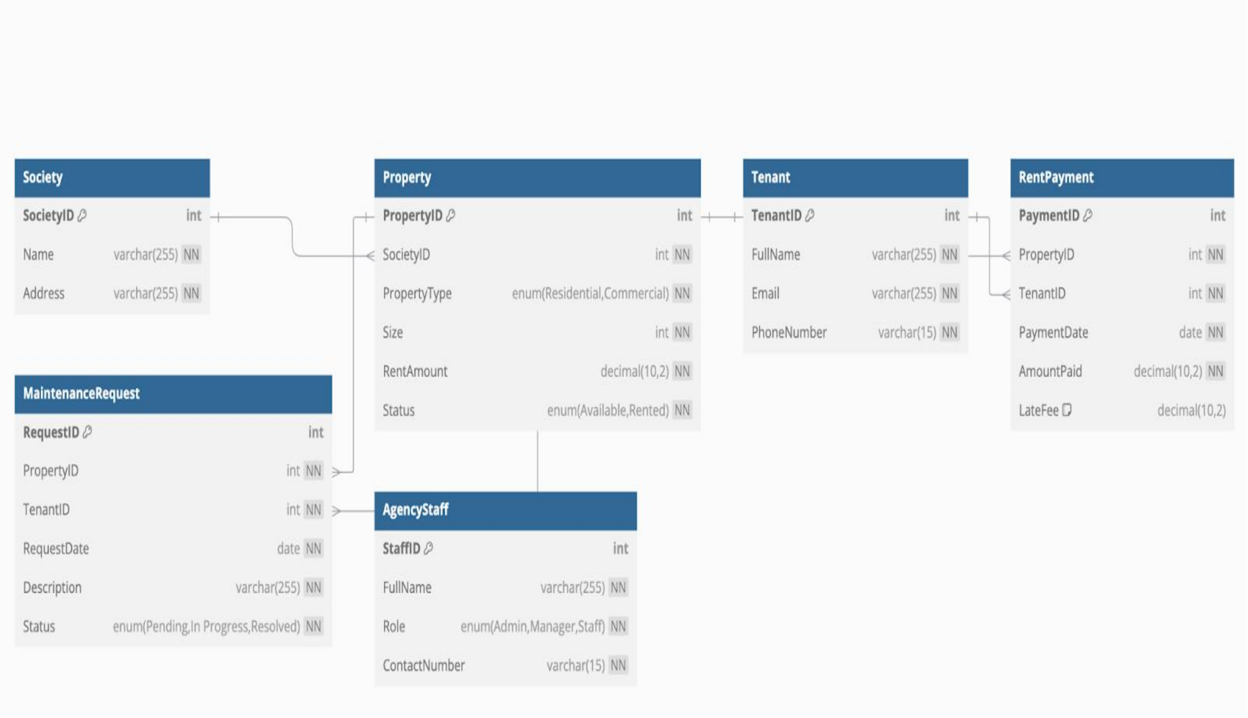
ERD and Relational Models

Entity-Relationship Diagram (ERD)

The ERD represents the entities, their attributes, and the relationships among them. Below is a simplified description of the entities in the Housing Management System:

- **Society**: Represents housing societies managed by the agency. It includes attributes like `SocietyID`, `Name`, and `Address`.
- **Property**: Represents properties within a society. Each property has attributes like `PropertyID`, `SocietyID`, `PropertyType`, `Size`, `RentAmount`, and `Status`.
- **Tenant**: Represents tenants renting properties. Includes `TenantID`, `FullName`, `Email`, and `PhoneNumber`.

- **RentPayment**: Represents rent payments made by tenants. Contains attributes like `PaymentID`, `PropertyID`, `TenantID`, `PaymentDate`, `AmountPaid`, and `LateFee`.
- **MaintenanceRequest**: Represents maintenance requests by tenants for properties. It includes `RequestID`, `PropertyID`, `TenantID`, `RequestDate`, `Description`, and `Status`.
- **AgencyStaff**: Represents staff members working in the housing agency. Includes attributes like `StaffID`, `FullName`, `Role`, and `ContactNumber`



SQL Code

DDL

```
CREATE DATABASE MISTP;
USE MISTP;

CREATE TABLE Society (
    SocietyID INT AUTO_INCREMENT PRIMARY KEY,
    Name NVARCHAR(255) NOT NULL,
    Address NVARCHAR(255) NOT NULL
);
```

```
-- Property Table
CREATE TABLE Property (
    PropertyID INT AUTO_INCREMENT PRIMARY KEY,
    SocietyID INT NOT NULL,
    PropertyType ENUM('Residential', 'Commercial') NOT NULL,
    Size INT NOT NULL,
    RentAmount DECIMAL(10, 2) NOT NULL,
    Status ENUM('Available', 'Rented') NOT NULL,
    FOREIGN KEY (SocietyID) REFERENCES Society(SocietyID)
);

-- Tenant Table
CREATE TABLE Tenant (
    TenantID INT AUTO_INCREMENT PRIMARY KEY,
    FullName NVARCHAR(255) NOT NULL,
    Email NVARCHAR(255) NOT NULL UNIQUE,
    PhoneNumber NVARCHAR(15) NOT NULL UNIQUE
);

-- RentPayment Table
CREATE TABLE RentPayment (
    PaymentID INT AUTO_INCREMENT PRIMARY KEY,
    PropertyID INT NOT NULL,
    TenantID INT NOT NULL,
    PaymentDate DATE NOT NULL,
    AmountPaid DECIMAL(10, 2) NOT NULL,
    LateFee DECIMAL(10, 2) DEFAULT 0,
    FOREIGN KEY (PropertyID) REFERENCES Property(PropertyID),
    FOREIGN KEY (TenantID) REFERENCES Tenant(TenantID)
);

-- MaintenanceRequest Table
CREATE TABLE MaintenanceRequest (
    RequestID INT AUTO_INCREMENT PRIMARY KEY,
    PropertyID INT NOT NULL,
    TenantID INT NOT NULL,
    RequestDate DATE NOT NULL,
    Description NVARCHAR(255) NOT NULL,
    Status ENUM('Pending', 'In Progress', 'Resolved') NOT NULL,
    FOREIGN KEY (PropertyID) REFERENCES Property(PropertyID),
    FOREIGN KEY (TenantID) REFERENCES Tenant(TenantID)
);

-- AgencyStaff Table
CREATE TABLE AgencyStaff (
    StaffID INT AUTO_INCREMENT PRIMARY KEY,
    FullName NVARCHAR(255) NOT NULL,
    Role ENUM('Admin', 'Manager', 'Staff') NOT NULL,
    ContactNumber NVARCHAR(15) NOT NULL
);
```

```sql
-- Index for faster lookup by TenantID in RentPayment
CREATE INDEX idx_tenantid_rentpayment ON RentPayment(TenantID);
-- Index for faster lookup by PropertyID in MaintenanceRequest
CREATE INDEX idx_propertyid_maintenance ON
MaintenanceRequest(PropertyID);
```

DML

```sql
-- Insert Data for Society
INSERT INTO Society (Name, Address)
VALUES
('Riverdale Apartments', '321 Maple Drive, Smallville'),
('Willow Creek Homes', '654 Cedar Lane, Star City'),
('Evergreen Villas', '111 Birch Boulevard, Central City'),
('Pine Valley Estates', '222 Redwood Road, Coast City'),
('Silver Oaks', '333 Aspen Avenue, Blüdhaven'),
('Crescent Hill Society', '444 Laurel Lane, Midway City'),
('Golden Meadows', '555 Walnut Street, Keystone City'),
('Sunset Grove', '123 Sunset Drive, Hill Valley'),
('Oceanview Residence', '456 Beach Road, Seaside City'),
('Mountain Ridge', '789 Highland Lane, Rocky Mountains'),
('Lakeside Village', '101 Lakeview Road, Clearwater'),
('Maplewood Apartments', '202 Maple Street, Autumn Town'),
('Hillcrest Enclave', '303 Hillside Avenue, Green Valley'),
('Birchwood Court', '404 Birchwood Drive, Red River'),
('Bluebell Manor', '505 Bluebell Way, Silver Creek');
```

```sql
-- Insert Data for Property
INSERT INTO Property (SocietyID, PropertyType, Size, RentAmount,
Status)

VALUES
(1, 'Residential', 1200, 3500.00, 'Available'),
(2, 'Commercial', 1500, 5000.00, 'Rented'),
(3, 'Residential', 1300, 4000.00, 'Available'),
(4, 'Commercial', 1100, 3000.00, 'Rented'),
(5, 'Residential', 1400, 4200.00, 'Available'),
(6, 'Residential', 1200, 3600.00, 'Rented'),
(7, 'Commercial', 1500, 5500.00, 'Available'),
(8, 'Residential', 1300, 3700.00, 'Rented'),
(9, 'Residential', 1100, 3000.00, 'Available'),
(10, 'Commercial', 1200, 5000.00, 'Rented'),
(11, 'Residential', 1400, 4500.00, 'Available'),
(12, 'Commercial', 1300, 4700.00, 'Rented'),
(13, 'Residential', 1600, 5500.00, 'Available'),
(14, 'Residential', 1100, 2900.00, 'Available'),
(15, 'Commercial', 1500, 5100.00, 'Rented');
```

-- Insert Data for Tenant

```sql
INSERT INTO Tenant (FullName, Email, PhoneNumber)
VALUES
('Alice Johnson', 'alice.johnson@example.com', '123-456-7890'),
('Bob Smith', 'bob.smith@example.com', '987-654-3210'),
('Charlie Brown', 'charlie.brown@example.com', '555-666-7777'),
('Diana Prince', 'diana.prince@example.com', '444-333-2222'),
('Edward Cullen', 'edward.cullen@example.com', '111-222-3333'),
('Fiona Adams', 'fiona.adams@example.com', '888-999-0000'),
('George Taylor', 'george.taylor@example.com', '777-888-9999'),
('Hannah Lee', 'hannah.lee@example.com', '666-777-8888'),
('Isaac Newton', 'isaac.newton@example.com', '555-444-3333'),
('Julia Roberts', 'julia.roberts@example.com', '444-555-6666'),
('Kevin Hart', 'kevin.hart@example.com', '333-444-5555'),
('Laura Hill', 'laura.hill@example.com', '222-333-4444'),
('Michael Jordan', 'michael.jordan@example.com', '999-888-7777'),
('Nancy Green', 'nancy.green@example.com', '111-000-9999'),
('Olivia White', 'olivia.white@example.com', '555-123-4567');
```

-- Insert Data for RentPayment

```sql
INSERT INTO RentPayment (PropertyID, TenantID, PaymentDate,
AmountPaid, LateFee)
VALUES
(2, 1, '2024-12-01', 3000.00, 0),
(4, 2, '2024-12-02', 1400.00, 50.00),
(6, 3, '2024-12-03', 3500.00, 0),
(8, 4, '2024-12-04', 1300.00, 0),
(10, 5, '2024-12-05', 1600.00, 100.00),
(1, 6, '2024-12-06', 1500.00, 0),
(3, 7, '2024-12-07', 1800.00, 0),
(5, 8, '2024-12-08', 1200.00, 50.00),
(7, 9, '2024-12-09', 1550.00, 0),
(9, 10, '2024-12-10', 1350.00, 0),
(11, 11, '2024-12-11', 3600.00, 50.00),
(12, 12, '2024-12-12', 1200.00, 0),
(13, 13, '2024-12-13', 1550.00, 25.00),
(14, 14, '2024-12-14', 2900.00, 0),
(15, 15, '2024-12-15', 5100.00, 10.00);
```

-- Insert Data for MaintenanceRequest

```sql
INSERT INTO MaintenanceRequest (PropertyID, TenantID, RequestDate,
Description, Status)
VALUES
(2, 1, '2024-11-28', 'Leaking faucet in kitchen', 'Resolved'),
(4, 2, '2024-12-01', 'Broken window in living room', 'In Progress'),
(6, 3, '2024-12-02', 'Heating system not working', 'Pending'),
(8, 4, '2024-12-03', 'Air conditioning malfunction', 'Resolved'),
(10, 5, '2024-12-04', 'Broken door lock', 'Pending'),
(1, 6, '2024-12-05', 'Clogged sink', 'In Progress'),
(3, 7, '2024-12-06', 'Faulty electrical wiring', 'Pending'),
(5, 8, '2024-12-07', 'Pest control issue', 'Resolved'),
(7, 9, '2024-12-08', 'Water heater not working', 'Resolved'),
(9, 10, '2024-12-09', 'Cracked tiles in bathroom', 'Pending'),
(11, 11, '2024-12-10', 'Broken kitchen cabinet', 'Resolved'),
(12, 12, '2024-12-11', 'Toilet flush not working', 'In Progress'),
(13, 13, '2024-12-12', 'Ceiling leak in bedroom', 'Pending'),
(14, 14, '2024-12-13', 'Faulty plumbing in bathroom', 'In Progress'),
(15, 15, '2024-12-14', 'Broken refrigerator door', 'Resolved');
```

-- Insert Data for AgencyStaff

```
INSERT INTO AgencyStaff (FullName, Role, ContactNumber)
VALUES
('John Doe', 'Admin', '222-333-4444'),
('Jane Smith', 'Manager', '555-444-3333'),
('Emily Davis', 'Staff', '666-555-4444'),
('Robert Johnson', 'Staff', '777-666-5555'),
('Samantha Carter', 'Manager', '888-777-6666'),
('William Moore', 'Admin', '999-888-7777'),
('Olivia Brown', 'Staff', '111-999-8888'),
('James White', 'Manager', '222-111-9999'),
('Sophia Clark', 'Staff', '333-222-1111'),
('Benjamin Adams', 'Admin', '444-333-2222'),
('Isabelle Green', 'Staff', '555-444-2222'),
('Daniel Harris', 'Manager', '666-555-3333'),
('Lucas King', 'Staff', '777-666-4444'),
('Maria Lewis', 'Admin', '888-777-5555'),
('Zoe Mitchell', 'Manager', '999-888-6666');
```

TRIGGER

-- Logging Table

```
CREATE TABLE RentPaymentLog (
    LogID INT AUTO_INCREMENT PRIMARY KEY,
    PaymentID INT,
    OldAmount DECIMAL(10,2),
    NewAmount DECIMAL(10,2),
    ChangeDate TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

```
CREATE TABLE RentPaymentLog (
    LogID INT AUTO_INCREMENT PRIMARY KEY,
    PaymentID INT,
    OldAmount DECIMAL(10,2),
    NewAmount DECIMAL(10,2),
    ChangeDate TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

```
DELIMITER $$

CREATE TRIGGER LogRentPaymentUpdate
AFTER UPDATE ON RentPayment
FOR EACH ROW
BEGIN
    INSERT INTO RentPaymentLog (PaymentID, OldAmount, NewAmount)
    VALUES (OLD.PaymentID, OLD.AmountPaid, NEW.AmountPaid);
END $$

DELIMITER ;

EXPLAIN SELECT * FROM RentPayment WHERE TenantID = 1;
SELECT * FROM RentedProperties;
UPDATE RentPayment SET AmountPaid = 2000 WHERE PaymentID = 1;
SELECT * FROM RentPaymentLog;
CALL TotalRentBySociety(1);
```

STORED PROCEDURE

```
DELIMITER //
CREATE PROCEDURE TotalRentBySociety(IN SocietyIDInput INT)
BEGIN
    SELECT
        s.Name AS SocietyName,
        SUM(rp.AmountPaid) AS TotalRentCollected
    FROM
        Society s
    JOIN
        Property p ON s.SocietyID = p.SocietyID
    JOIN
        RentPayment rp ON p.PropertyID = rp.PropertyID
    WHERE
        s.SocietyID = SocietyIDInput
    GROUP BY
        s.Name;
DELIMITER ;
```

VIEW

```
-- View for Properties Rented
CREATE VIEW RentedProperties AS
SELECT
    p.PropertyID,
    s.Name AS SocietyName,
    p.PropertyType,
    p.RentAmount,
    t.FullName AS TenantName
FROM
```

```
    Property p
JOIN
    Society s ON p.SocietyID = s.SocietyID
JOIN
    RentPayment rp ON p.PropertyID = rp.PropertyID
JOIN
    Tenant t ON rp.TenantID = t.TenantID
WHERE
    p.Status = 'Rented';
```

Database Deployment Documentation

To deploy the database on AWS (Amazon Web Services):

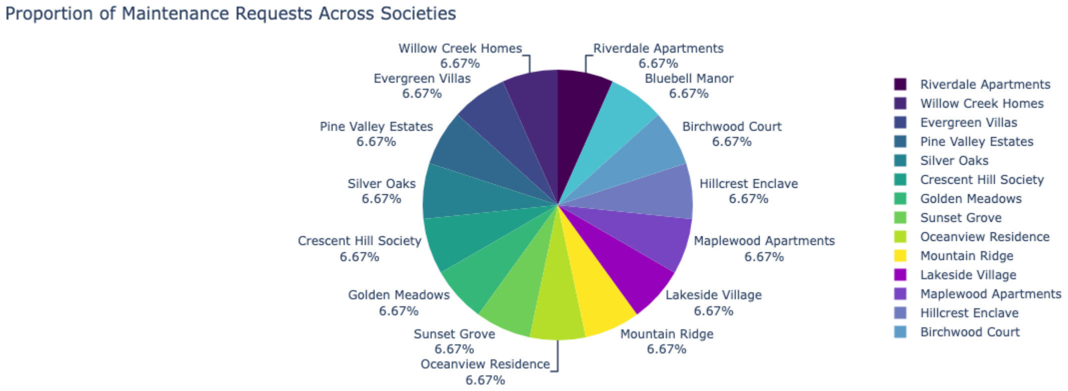
1. **Choose Amazon RDS**:
    a. Select **Amazon RDS** for a relational database service.
    b. Choose MySQL or PostgreSQL based on your preference.
    c. Launch the database instance in the free tier, selecting configurations for your application.
2. **Security and Access**:
    a. Set up **security groups** to define who can access the database (e.g., IP address or VPC).
    b. Create **database users** with different access levels:
        i. Admin: Full access
        ii. Data Entry: Limited access to add or modify data
        iii. Read-Only: Read-only access for reporting
3. **Backup and Maintenance**:
    a. Enable automated backups for disaster recovery.
    b. Set up maintenance schedules for system updates and patches.

Dashboard with explanations for the analytical insights

1. Which society has the highest number of maintenance requests?

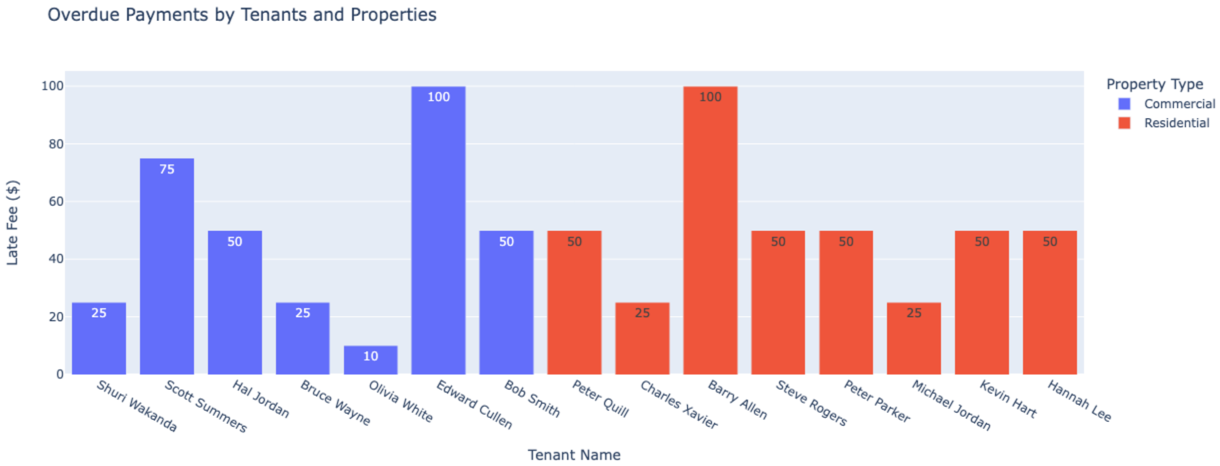
- **Tables Involved**: `Society, Property, MaintenanceRequest`

- **Explanation**: Maintenance requests are tied to properties, and properties belong to societies. A query joining these tables can aggregate requests by society.



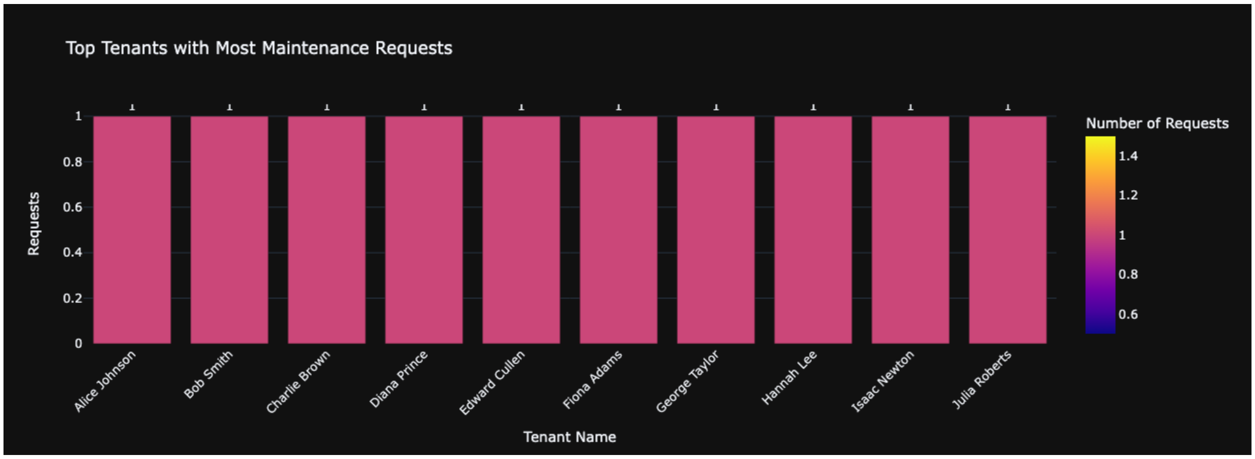
2. Which properties have overdue payments?

- **Tables Involved**: `RentPayment, Property, Tenant`
- **Explanation**: Late fees in the `RentPayment` table can indicate overdue payments. A query can identify properties with non-zero late fees.



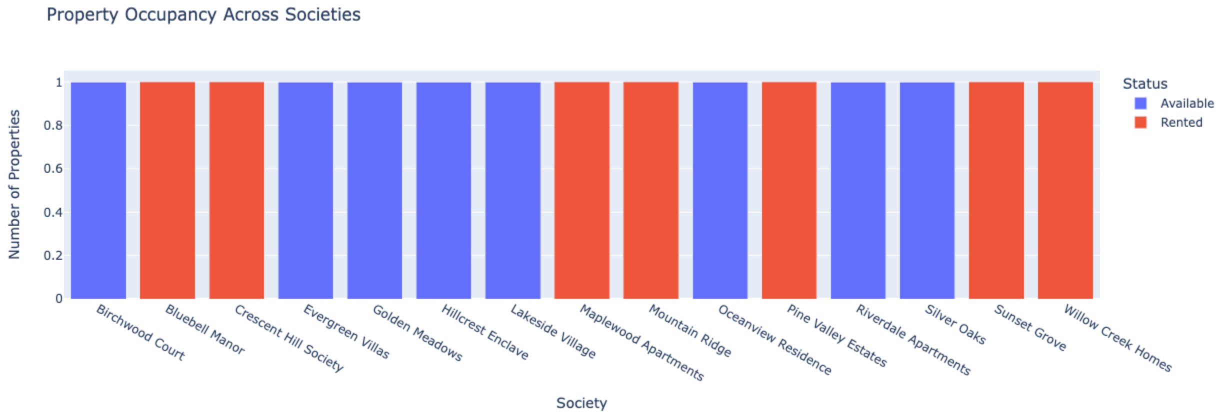
3. Which tenants have the most maintenance requests?

- **Tables Involved**: `Tenant, MaintenanceRequest`
- **Explanation**: Maintenance requests are linked to tenants. Aggregating the number of requests by tenant gives the desired insight.



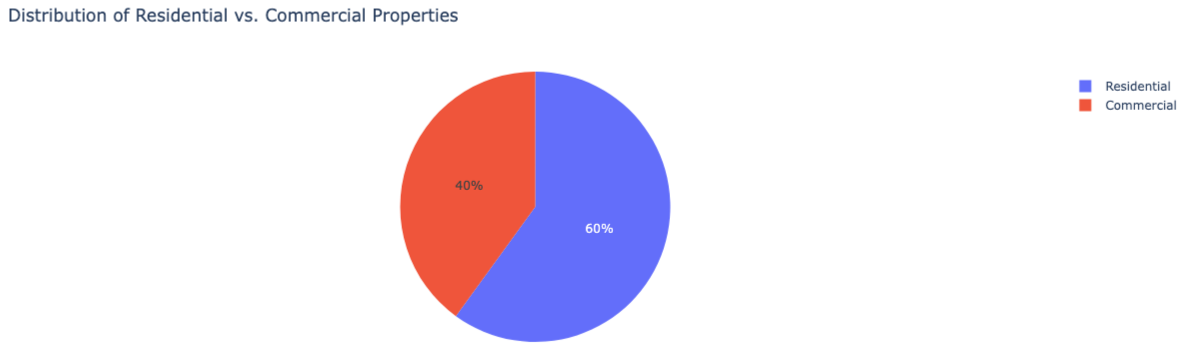
4. How does property occupancy vary across societies?

- **Tables Involved**: `Society, Property`
- **Explanation**: The Status field in the `Property` table indicates whether a property is rented. Grouping by society and counting `Rented` vs. `Available` properties reveals occupancy trends.



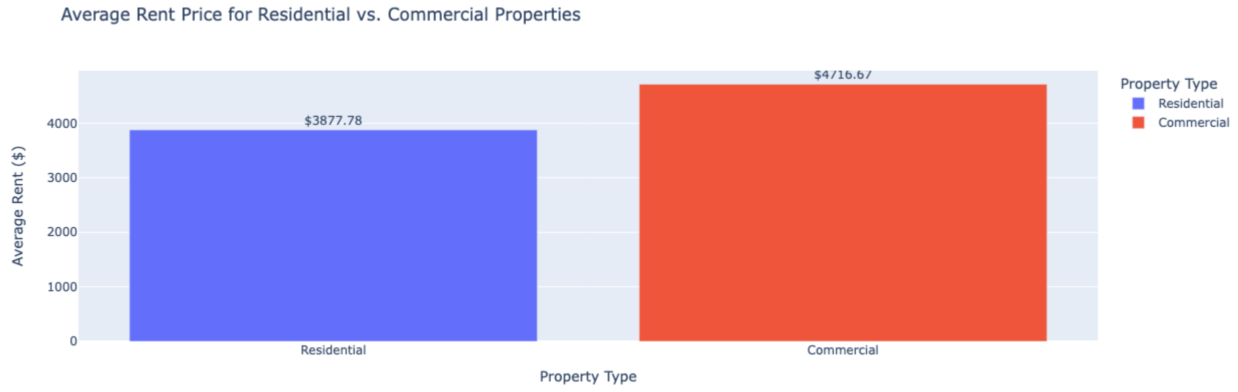
5. What is the distribution of residential vs. commercial properties?

- **Tables Involved:** `Property`
- **Explanation**: The `PropertyType` field in the `Property` table enables a breakdown of residential vs. commercial properties.



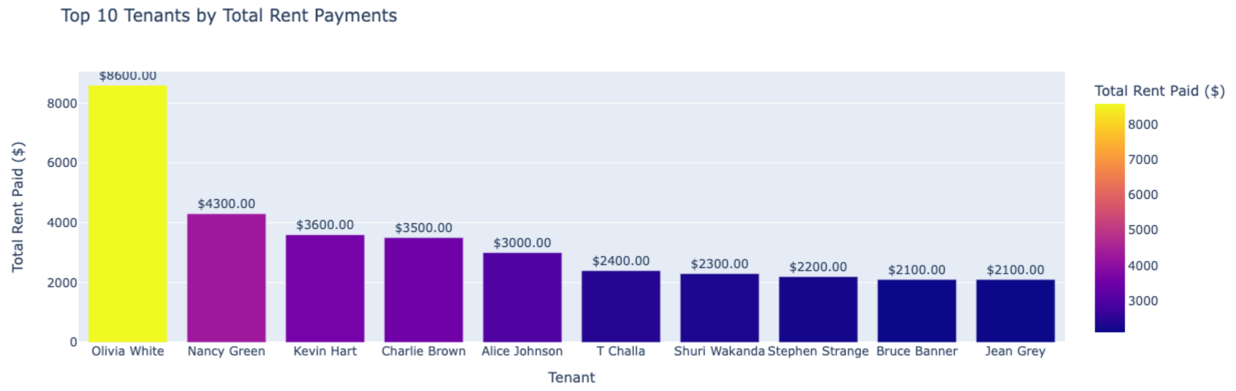
6. What is the average rent price for residential vs. commercial properties?

- **Tables Involved**: `Property`
- **Explanation**: You can calculate the average rent for both property types (residential and commercial) by grouping the properties by type and averaging the `RentAmount`.



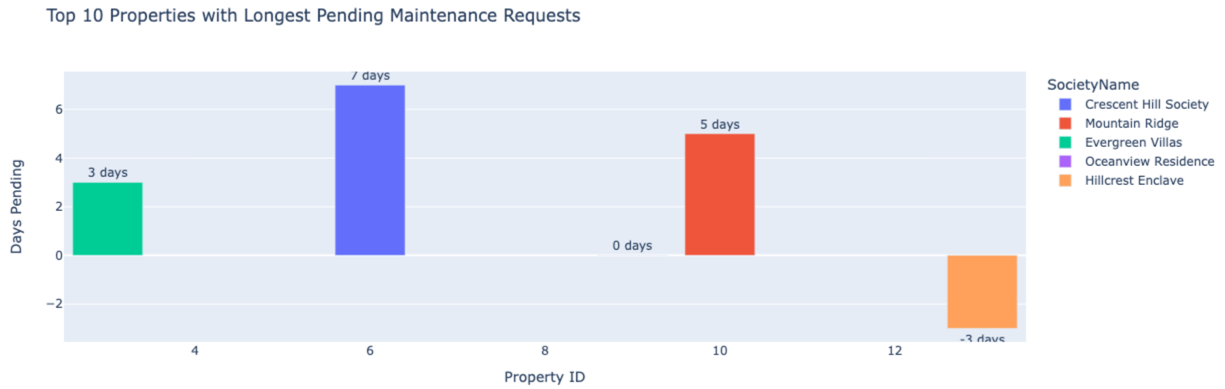
7. Which tenants have made the highest rent payments?

- **Tables Involved**: `Tenant, RentPayment`
- **Explanation**: Aggregating the `AmountPaid` for each tenant will allow you to identify which tenants have made the highest payments.



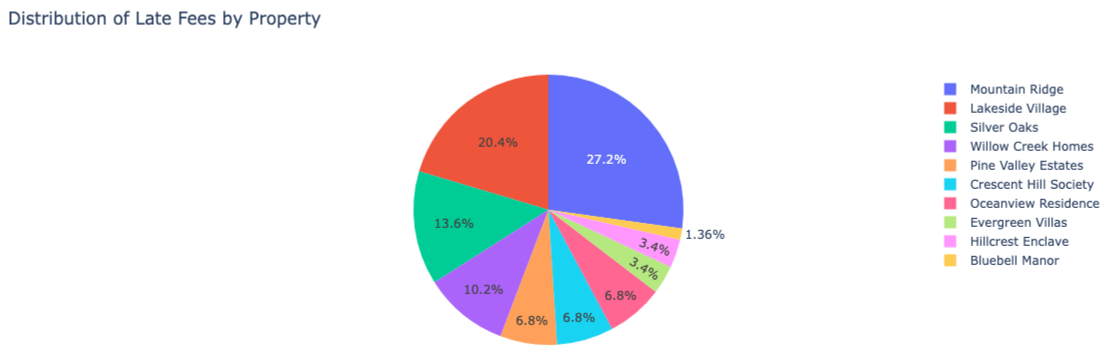
8. Which properties have the longest pending maintenance requests?

- **Tables Involved**: `Property, MaintenanceRequest`
- **Explanation**: By filtering maintenance requests with a "Pending" status and calculating the time difference between `RequestDate` and the current date, you can identify which properties have the longest pending maintenance requests.



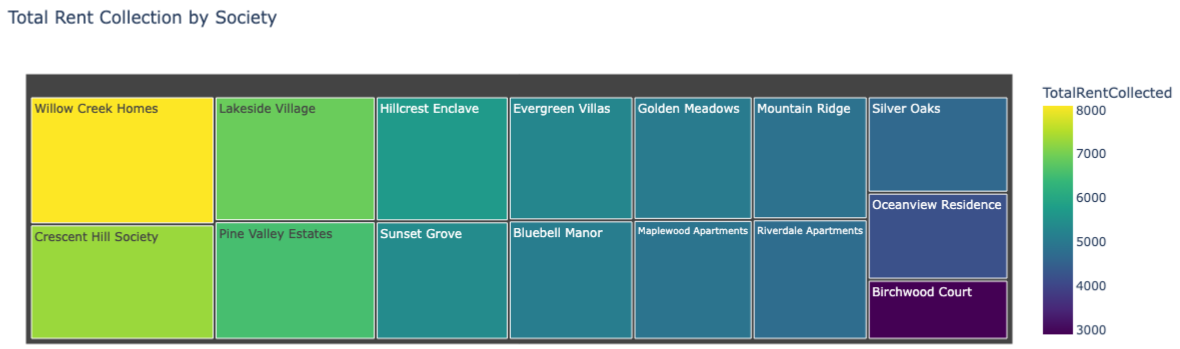
9. What is the total late fee generated by overdue payments across all properties?

- **Tables Involved**: `RentPayment`
- **Explanation**: Summing the `LateFee` column in the `RentPayment` table gives the total late fee generated for all overdue payments.



10. Which society has the highest total rent collection?

- **Tables Involved**: `Society, Property, RentPayment`
- **Explanation**: Summing the rent payments for properties within each society will allow you to determine which society has the highest rent collection.


Total Rent Collection by Society
Willow Creek Homes
Lakeside Village
Hillcrest Enclave
Evergreen Villas
Golden Meadows
Mountain Ridge
Silver Oaks
Crescent Hill Society
Pine Valley Estates
Sunset Grove
Bluebell Manor
Maplewood Apartments
Riverdale Apartments
Oceanview Residence
Birchwood Court
TotalRentCollected