Group-09 Maintenance Request Control Database Management System

Sowmya Vangalapudi (Team Coordinator)

Rajiv Adusumalli

Tarun Podila

Varun Kumar Atkuri

INFO 5707 Data Modeling for Information Professionals

Dr. Tozammel Hossain

Contents

Objective:

The main goal of this database is to make it easier and more efficient to handle maintenance requests. By offering a system the aim is to simplify the process of submitting, assigning and tracking maintenance tasks to ensure they get resolved promptly. Additionally, through reporting and analysis tools the system aims to help make decisions and enhance operations for improved efficiency and user happiness.

Scope:

The scope of this system is to build an efficient maintenance system for industries like real estate and property management, educational institutions, facilities managements. It primarily focuses on processing the maintenance orders and request tracking at department/task level at these organizations, while also promoting collaboration among different parties engaged in the maintenance activities.

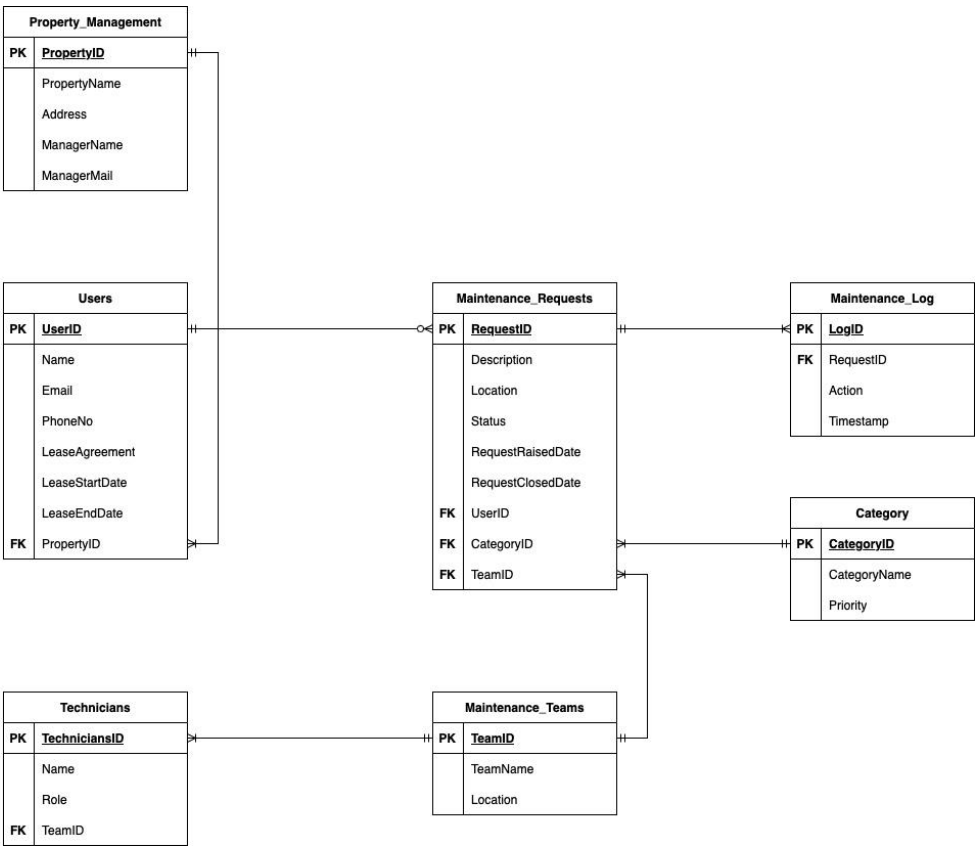
User Requirements:

1. Every property must be managed by a manager.
2. Users/tenants should be able to easily submit maintenance requests, providing details like problem category, location, and urgency, through an intuitive interface.
3. Administrators should be able to assign requests to maintenance teams and share the details duties, including location, issue description, and priority.
4. Users should provide additional context or information related to their maintenance requests as needed.
5. Requests should be categorized and prioritized based on severity and impact on operations to ensure critical issues are addressed promptly.
6. Users should be able to track their maintenance request status updates.
7. Maintenance requests must include dates indicating when they were raised and completed.
8. Every action regarding the request should be logged.
9. Users and property managers should have access to historical data and records of their past maintenance activities.
10. The interface for submitting maintenance requests should be user-friendly and intuitive to ensure ease of use for all users.
11. Users should be able to view any changes or updates to their maintenance requests.
12. The system should provide a clear and transparent process for users to address their requests.

Business Rules:

1. Every property has a manager to administrate the property.
2. Every property will have multiple tenants, but each tenant is assigned to one property.
3. Multiple tenants can raise maintenance requests, but each request is associated with only one tenant.
4. Every request should be segregated based on the category/problem faced by the user/tenant.
5. Each maintenance team is specialized in one category/problem.
6. Based on the request category, the administrator should assign them to respective maintenance team.
7. Each request is assigned to a single maintenance team, but each team can handle many requests.
8. The maintenance team should assign technicians to resolve the request.
9. Each technician works only in one maintenance team, but maintenance teams have multiple technicians.
10. Maintenance teams should be able to view the duties, including location, issue description, and priority.
11. The request's status can have multiple stages like open when it is created, in progress when assigned to a technician and completed when completed.
12. The request should have the dates of request raised and request completed.
13. The system must maintain a log of all actions performed.

ER Diagram:


Property_Management
PK PropertyID
PropertyName
Address
ManagerName
ManagerMail
Users
PK UserID
Name
Email
PhoneNo
LeaseAgreement
LeaseStartDate
LeaseEndDate
FK PropertyID
Maintenance_Requests
PK RequestID
Description
Location
Status
RequestRaisedDate
RequestClosedDate
FK UserID
FK CategoryID
FK TeamID
Maintenance_Log
PK LogID
FK RequestID
Action
Timestamp
Category
PK CategoryID
CategoryName
Priority
Technicians
PK TechniciansID
Name
Role
FK TeamID
Maintenance_Teams
PK TeamID
TeamName
Location

Assumptions:

1. Property_Management and Users entities have 1:M relation, as the property can have multiple tenants, but each tenant belongs to one property.
2. Users and Maintenance_Requests entities have 1:0* relation, as each user can have zero or multiple requests, but each request is raised a user.
3. Each Maintenance team is assigned to multiple requests, but one request is only tagged to one maintenance team.
4. TeamName attribute in the Maintenance_teams entity represents the specialization of that whole team i.e., “Plumbing team”, “Electrical team”.
5. Priority attribute in the category entity will have the value Low, Medium, High varies with category name attribute.
6. Timestamp attribute of the Maintenance_Log entity have the datatype DATETIME rather than TIMESTAMP as it has wider range values.
7. Category and Maintenance_Requests entities have 1:M relation, as each category can be used in multiple requests, but each request has single category.

Entities:

1. Property_Management: PropertyID [PK], PropertyName, Address, ManagerName, ManagerMail
2. Users: UserID [PK], Name, Email, PhoneNo, LeaseAgreement, LeaseStartDate, LeaseEndDate, PropertyID [FK]
3. Maintenance_Requests: RequestID [PK], UserID [FK], CategoryID [FK], TeamID [FK], RequestRaisedDate, RequestCompletedDate, Status, Address, Description
4. Category: CategoryID [PK], CategoryName, Priority
5. Maintenance_Teams: TeamID [PK], TeamName, Location
6. Technicians: TeachnicianID [PK], TeamID [FK], Name, Role
7. Maintenance_Log: LogID [PK], RequestID [FK], Action, Timestamp

Data Dictionary:

Table Name	Attribute Name	Description	Data Type	Data Format	Required	PK or FK	Example
Property_Management	PropertyID	Unique identifier for each property	INT	XXXX XXXX XX	Y	PK	123456
	PropertyName	Name/Title of the property	VARCHAR (50)	XXXX XXXX	Y		"VENUE Apartments"
	Address	Address of the property	VARCHAR (255)	XXX, 123	Y		"123 Main Street, DENTON, TX-76201"
	ManagerName	Name of the property manager	VARCHAR (20)	XXXX XXXX	Y		"John Grisham"
	ManagerMail	Email of the property manager	VARCHAR (25)	abc@domain.com	Y		john.grisham@gmail.com
Users	UserID	Unique identifier for each user/tenant	INT	&&&& &&&& &	Y	PK	1234560001
	Name	Name of the user/tenant	VARCHAR (20)	XXXX XXXX	Y		"Bradd Pitt"
	Email	Email address of the user/tenant	VARCHAR (25)	abc@domain.com	Y		bradd.pitt@gmail.com
	PhoneNo	Phone number of the user/tenant	INT	XXXXX XXX	N		"123-456-7890"

	LeaseAgreement	Information related to the lease agreement	VARCHAR (50)	XXX, 123	N		"Lease #1234560001 "
	LeaseStartDate	Start date of the lease agreement	DATE	YYYY-MM-DD	N		"2022-01-01"
	LeaseEndDate	End date of the lease agreement	DATE	YYYY-MM-DD	N		"2022-12-31"
	PropertyID	Foreign key referencing Property_Management	INT	XXXX XXXX XX	Y	FK	123456
Maintenance_ Requests	RequestID	Unique identifier for each maintenance request	INT	XXXX XXXX X	Y	PK	110022
	UserID	Foreign key referencing Users	INT	&&&& &&&& &	Y	FK	1234560001
	CategoryID	Foreign key referencing Category	INT	AAAAA	Y	FK	301
	TeamID	Foreign key referencing Maintenance_ Teams	INT	VVVVV V	Y	FK	121001
	RequestRaisedDate	Date when the request was raised	DATE	YYYY-MM-DD	Y		"2022-03-15"

	RequestCompletedDate	Date when the request was completed	DATE	YYYY-MM-DD	N		"2022-03-20"
	Status	Status of the request	VARCHAR (15)	SSSSS	Y		"Open"
	Location	Location of the maintenance issue	VARCHAR (255)	XXX, 123	Y		"123 Main Street, Apt 112, DENTON, TX-76201"
	Description	Description of the maintenance issue	VARCHAR (255)	abc, 123	Y		"Leaking faucet"
Category	CategoryID	Unique identifier for each category of maintenance	INT	AAAAA	Y	PK	301
	CategoryName	Name of the maintenance category	VARCHAR (20)	CCCCC	Y		"Plumbing"
	Priority	Level of priority of the request	VARCHAR (10)	PPPPP	Y		"High"
Maintenance_Teams	TeamID	Unique identifier for each maintenance team	INT	VVVVVV	Y	PK	121001
	TeamName	Name of the maintenance team	VARCHAR (20)	XXXXX	Y		"Plumbing Team"

	Location	Location of the maintenance team	VARCHAR (255)	XXX, 123	Y		"Building A, Floor 1, Denton,TX"
Technicians	TechnicianID	Unique identifier for each technician	INT	YYYY YY	Y	PK	501
	TeamID	Foreign key referencing Maintenance_ Teams	INT	VVVVV V	Y	FK	121001
	Name	Name of the technician	VARCHAR (20)	XXXX	Y		"Steve Smith"
	Role	Role of the technician	VARCHAR (15)	XXXX XX	N		"Plumber"
Maintenance_ Log	LogID	Unique identifier for each log entry	INT	XXXX X	Y	PK	601
	RequestID	Foreign key referencing Maintenance_ Requests	INT	XXXX XXXX X	Y	FK	110022
	Action	Description of the action performed	VARCHAR (255)	abc, 123	Y		"Assigned to Technician"
	Timestamp	Timestamp indicating when the action was performed	DATETIME	YYYY-MM-DD HH:MM :SS	Y		"2022-03-15 09:00:00"

PART-2

Section1: Entity Generation and Data Entry:

Entity Generation: Here we have basically created a schema called MaintainEase and we are using it to create entities.

The tables created were Property_Management, Users, Category, Maintenance_Teams, Technicians, Maintenance_Requests and Maintenance_Logs. These tables were created by making use of CREATE Statements which will be used to define their properties with data types.

We have included the statements we have used for entity generation below and the reasons why we created those entities. Below are the commands:

1)We need a schema to store our data by creating tables in it.

```
CREATE SCHEMA MaintainEase;
```

2) We need to use the schema to create tables and entry data into those tables.

```
USE MaintainEase;
```

3) We need an entity to store the details of each property taken care of by the Management. This entity contains the property address and manager details of those properties.

```
-- Create Property_Management table
  CREATE TABLE Property_Management (
   PropertyID INT AUTO_INCREMENT PRIMARY KEY,
   PropertyName VARCHAR(50) NOT NULL,
   Address VARCHAR(255) NOT NULL, -- Changed from Location
   ManagerName VARCHAR(25) NOT NULL,
   ManagerMail VARCHAR(50) NOT NULL
);
```

4) We need an entity to store the details of every user of those properties. It stores the details of users and their propertyid in which they reside.

```
-- Create Users table
CREATE TABLE Users (
```

```
    UserID INT AUTO_INCREMENT PRIMARY KEY,
    Name VARCHAR(25) NOT NULL,
    Email VARCHAR(50) NOT NULL UNIQUE, -- Added UNIQUE constraint
    PhoneNo INT,
    LeaseAgreement VARCHAR(255),
    LeaseStartDate DATE,
    LeaseEndDate DATE,
    PropertyID INT,
    FOREIGN KEY (PropertyID) REFERENCES Property_Management(PropertyID)
);
```

5) As we are making a maintenance request system, we need to know the category of every request raised. So, we created a table for the category of each request they belong to. It stores the details of each category and its priority.

```
-- Create Category table
   CREATE TABLE Category (
    CategoryID INT AUTO_INCREMENT PRIMARY KEY,
    CategoryName VARCHAR(255) NOT NULL,
    Priority VARCHAR(10) NOT NULL CHECK (Priority IN ('Low', 'Medium', 'High')) -- Added CHECK
constraint
);
```

6) We need to know which team is assigned for each maintenance request raised. We created a maintenance team's entity to store the details of each team and their location.

```
-- Create Maintenance_Teams table
   CREATE TABLE Maintenance_Teams (
    TeamID INT AUTO_INCREMENT PRIMARY KEY,
    TeamName VARCHAR(25) NOT NULL,
    Location VARCHAR(255) NOT NULL
);
```

We need to have the team members' details of each team. So, we created a table which contains every technician who is part of every team and their role.

```
-- Create Technicians table
CREATE TABLE Technicians (
    TechnicianID INT AUTO_INCREMENT PRIMARY KEY,
    TeamID INT,
    Name VARCHAR(25) NOT NULL,
    Role VARCHAR(20),
    FOREIGN KEY (TeamID) REFERENCES Maintenance_Teams(TeamID)
);
```

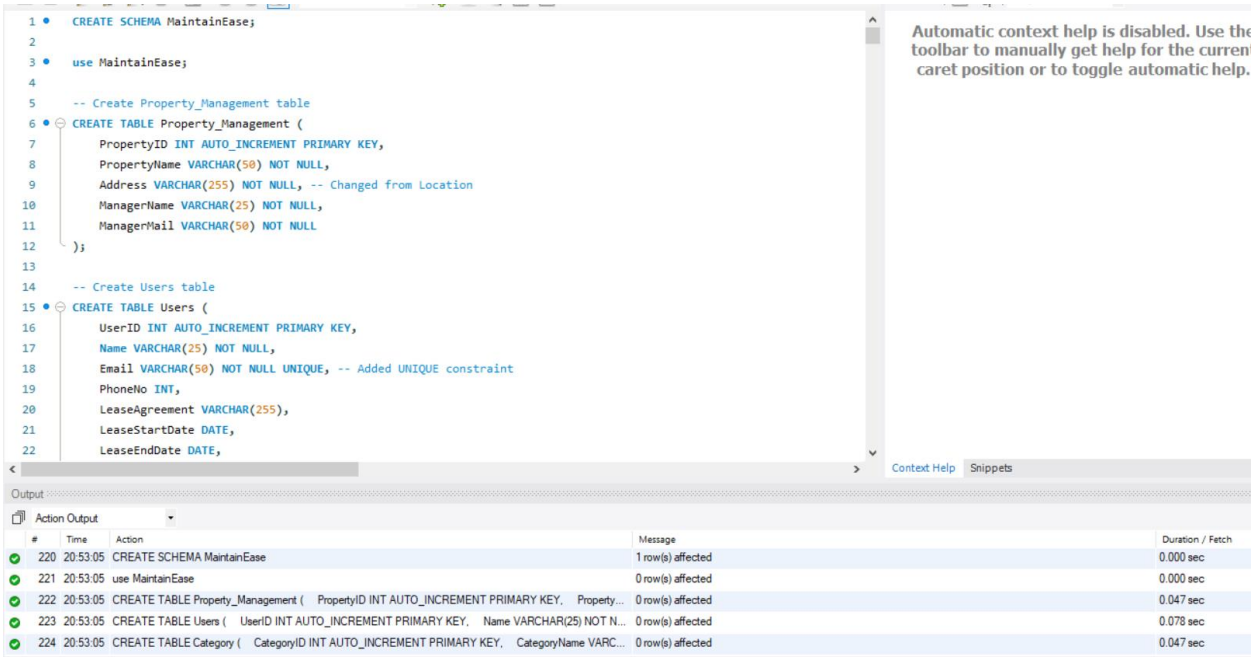
7) We need to know the details about each request raised by the user. So, we have created a table which contains every detail of a request like when it is raised, by which user is, what the request is about, and which team is assigned to resolve the request.

```
-- Create Maintenance_Requests table
CREATE TABLE Maintenance_Requests (
    RequestID INT AUTO_INCREMENT PRIMARY KEY,
    UserID INT,
    CategoryID INT,
    TeamID INT,
    RequestRaisedDate DATE NOT NULL,
    RequestCompletedDate DATE,
    Status VARCHAR(20) NOT NULL CHECK (Status IN ('Open', 'In Progress', 'Completed')), -- Added CHECK constraint
    Location VARCHAR(255) NOT NULL,
    Description TEXT NOT NULL,
    FOREIGN KEY (UserID) REFERENCES Users(UserID),
    FOREIGN KEY (CategoryID) REFERENCES Category(CategoryID),
    FOREIGN KEY (TeamID) REFERENCES Maintenance_Teams(TeamID)
);
```

8)We need to log every action taken to maintenance requests. So, we have created a table to store the logs of every action made to a request.

```
-- Create Maintenance_Log table

CREATE TABLE Maintenance_Log (

LogID INT AUTO_INCREMENT PRIMARY KEY,

RequestID INT,

Action VARCHAR(255) NOT NULL,

Timestamp DATETIME DEFAULT CURRENT_TIMESTAMP, -- Changed to DATETIME

FOREIGN KEY (RequestID) REFERENCES Maintenance_Requests(RequestID)

);
```

Screenshot of execution and output:



Data Entry: We have used INSERT commands for data entry into each entity. Let's see one by one.

1)Data Entry into **Property_Management** entity:

By using insert command and by giving the values for the respective attributes, the data is entered into Property_Management entity.

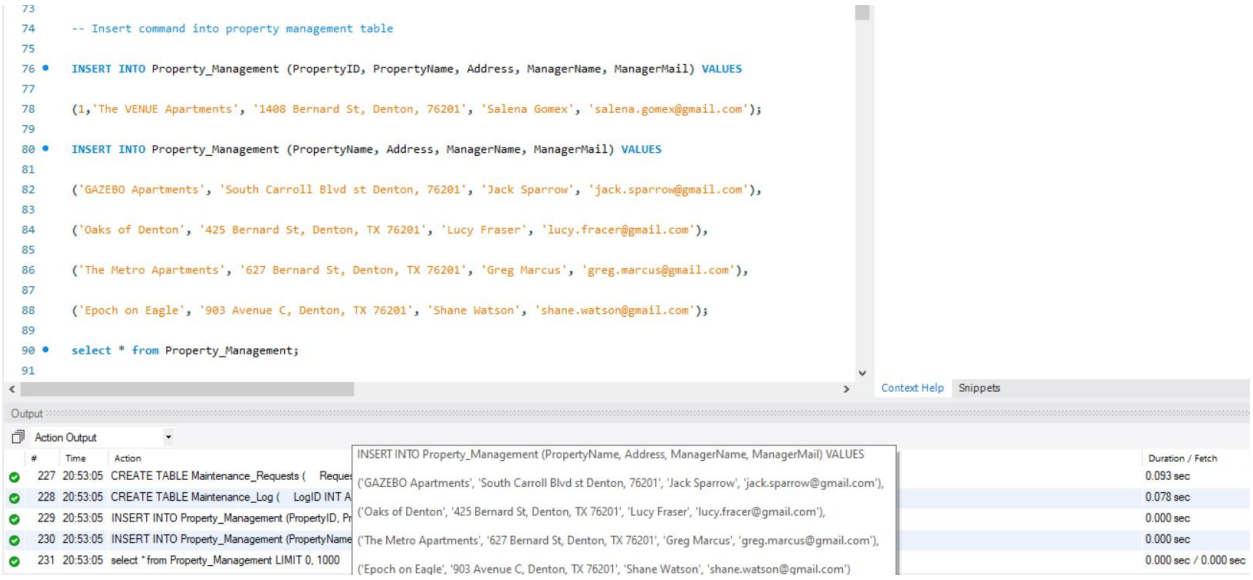
The PropertyID is actually given only for the first record.

Here as it is defined as auto increment attribute for the following next records the PropertyID will increment depending on the first record's PropertyID. And we have made sure we entered 5 number of records into the Property_Management entity as we made a presumption that MaintainEase properties only will control five properties.

-- Insert command into property management table

```
INSERT INTO Property_Management (PropertyID, PropertyName, Address, ManagerName, ManagerMail)
VALUES
(1,'The VENUE Apartments', '1408 Bernard St, Denton, 76201', 'Salena Gomex', 'salena.gomex@gmail.com');
INSERT INTO Property_Management (PropertyName, Address, ManagerName, ManagerMail) VALUES
('GAZEBO Apartments', 'South Carroll Blvd st Denton, 76201', 'Jack Sparrow', 'jack.sparrow@gmail.com'),
('Oaks of Denton', '425 Bernard St, Denton, TX 76201', 'Lucy Fraser', 'lucy.fracer@gmail.com'),
('The Metro Apartments', '627 Bernard St, Denton, TX 76201', 'Greg Marcus', 'greg.marcus@gmail.com'),
('Epoch on Eagle', '903 Avenue C, Denton, TX 76201', 'Shane Watson', 'shane.watson@gmail.com');
```

Below is the screenshot of the SQL insert commands and respective output.

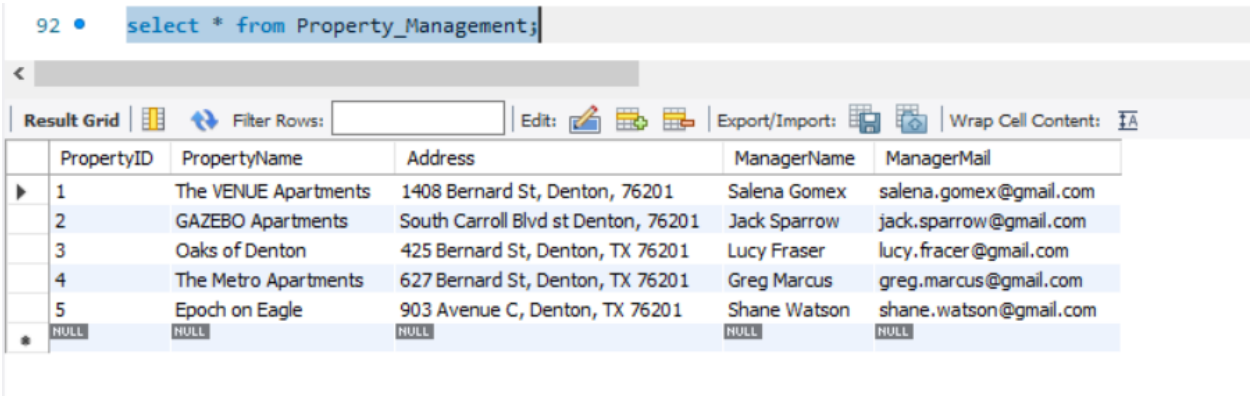


Using the below SQL command, we want to show the records of the Property_Management Entity.

SQL Command:

Select * from Property_Management;

Screenshot of command and its output:



2)Data Entry into **Users** entity:

Using the insert command again, data is added successfully to the Users object by giving the attribute values. Only for the first entry, the UserID is given as 1100001.

And we must note that the UserID for foloowing entries will be incremented depending upon the UserID of the first record, according to the definition of auto increment attribute.

-- Insert data into user table

INSERT INTO Users (UserID, Name, Email, PhoneNo, LeaseAgreement, LeaseStartDate, LeaseEndDate, PropertyID)

```
VALUES

(1100001,'Jessica Smith', 'jessicasmith@gmail.com', 1234567890, 'LEASE AGREEMENT-VA100', '2023-09-15', '2024-07-01', 1);

ALTER TABLE Users MODIFY PhoneNo BIGINT;

INSERT INTO Users (Name, Email, PhoneNo, LeaseAgreement, LeaseStartDate, LeaseEndDate, PropertyID)

VALUES

('Michael Johnson', 'michaeljohnson@gmail.com', 2345678901, 'LEASE AGREEMENT-VA101', '2023-10-20', '2024-09-10', 1),

('Emily Brown', 'emilybrown@gmail.com', 3456789012, 'LEASE AGREEMENT-VA102', '2023-08-05', '2024-12-15', 1),

('Christopher Davis', 'christopherdavis@gmail.com', 4567890123, 'LEASE AGREEMENT-VA103', '2023-06-25', '2024-11-20', 1),

('Sarah Wilson', 'sarahwilson@gmail.com', 5678901234, 'LEASE AGREEMENT-VA104', '2023-11-10', '2024-08-25', 1),

('Matthew Martinez', 'matthewmartinez@gmail.com', 6789012345, 'LEASE AGREEMENT-VA105', '2023-07-15', '2024-10-05', 1),

('Amanda Anderson', 'amandaanderson@gmail.com', 7890123456, 'LEASE AGREEMENT-VA106', '2023-05-30', '2024-09-30', 1),

('Daniel Taylor', 'danieltaylor@gmail.com', 8901234567, 'LEASE AGREEMENT-VA107', '2023-10-05', '2024-12-10', 1),

('Lauren Thomas', 'laurenthomas@gmail.com', 9012345678, 'LEASE AGREEMENT-VA108', '2023-06-15', '2024-10-20', 1),

('Ryan Jackson', 'ryanjackson@gmail.com', 1236786490, 'LEASE AGREEMENT-VA109', '2023-08-20', '2024-11-15', 1),

('Ashley White', 'ashleywhite@gmail.com', 2356789014, 'LEASE AGREEMENT-VA110', '2023-07-05', '2024-08-15', 1),

('John Harris', 'johnharris@gmail.com', 3478901256, 'LEASE AGREEMENT-VA111', '2023-09-10', '2024-09-05', 1),

('Megan Clark', 'meganclark@gmail.com', 4567012389, 'LEASE AGREEMENT-VA112', '2023-06-30', '2024-12-01', 1),

('Andrew Lewis', 'andrewlewis@gmail.com', 5690123478, 'LEASE AGREEMENT-VA113', '2023-11-05', '2024-08-10', 1),

('Olivia Lee', 'oliviale@gmail.com', 6782345901, 'LEASE AGREEMENT-VA114', '2023-08-10', '2024-10-15', 1),

('James Young', 'jamesyoung@gmail.com', 7123456890, 'LEASE AGREEMENT-VA115', '2023-05-20', '2024-11-30', 1),
```

('Jessica Turner', 'jessicaturner@gmail.com', 8034567912, 'LEASE AGREEMENT-VA116', '2023-10-15', '2024-09-20', 1),

('William Rodriguez', 'williamrodriguez@gmail.com', 9034578126, 'LEASE AGREEMENT-VA117', '2023-07-25', '2024-08-30', 1),

('Samantha Walker', 'samanthawalker@gmail.com', 1236790458, 'LEASE AGREEMENT-VA118', '2023-09-30', '2024-10-10', 1),

('Brandon Evans', 'brandonevans@gmail.com', 2457901368, 'LEASE AGREEMENT-VA119', '2023-08-05', '2024-12-05', 1),

('Jennifer Garcia', 'jennifergarcia@gmail.com', 1234678905, 'LEASE AGREEMENT-GA100', '2023-09-15', '2024-07-01', 2),

('David Martinez', 'davidmartinez@gmail.com', 2456789013, 'LEASE AGREEMENT-GA101', '2023-10-20', '2024-09-10', 2),

('Mary Lopez', 'marylopez@gmail.com', 3567890124, 'LEASE AGREEMENT-GA102', '2023-08-05', '2024-12-15', 2),

('Jose Gonzalez', 'josegonzalez@gmail.com', 4678901235, 'LEASE AGREEMENT-GA103', '2023-06-25', '2024-11-20', 2),

('Maria Rodriguez', 'mariarodriguez@gmail.com', 5789012346, 'LEASE AGREEMENT-GA104', '2023-11-10', '2024-08-25', 2),

('James Martinez', 'jamesmartinez@gmail.com', 6890123457, 'LEASE AGREEMENT-GA105', '2023-07-15', '2024-10-05', 2),

('Jessica Hernandez', 'jessicahernandez@gmail.com', 7901234568, 'LEASE AGREEMENT-GA106', '2023-05-30', '2024-09-30', 2),

('Daniel Garcia', 'danielgarcia@gmail.com', 8012345679, 'LEASE AGREEMENT-GA107', '2023-10-05', '2024-12-10', 2),

('Emma Lopez', 'emmalopez@gmail.com', 9023456781, 'LEASE AGREEMENT-GA108', '2023-06-15', '2024-10-20', 2),

('Alexander Martinez', 'alexandermartinez@gmail.com', 1235678904, 'LEASE AGREEMENT-GA109', '2023-08-20', '2024-11-15', 2),

('Sophia Hernandez', 'sophiahernandez@gmail.com', 2456780139, 'LEASE AGREEMENT-GA110', '2023-07-05', '2024-08-15', 2),

('Jacob Gonzalez', 'jacobgonzalez@gmail.com', 3468012597, 'LEASE AGREEMENT-GA111', '2023-09-10', '2024-09-05', 2),

('Isabella Rodriguez', 'isabellarodriguez@gmail.com', 4567823190, 'LEASE AGREEMENT-GA112', '2023-06-30', '2024-12-01', 2),

('William Garcia', 'williamgarcia@gmail.com', 5791234806, 'LEASE AGREEMENT-GA113', '2023-11-05', '2024-08-10', 2),

('Sophia Martinez', 'sophiamartinez@gmail.com', 6781245309, 'LEASE AGREEMENT-GA114', '2023-08-10', '2024-10-15', 2),

('Daniel Fernandez', 'danielfernandez@gmail.com', 7893456102, 'LEASE AGREEMENT-GA115', '2023-05-20', '2024-11-30', 2),

('Olivia Lopez', 'olivialopez@gmail.com', 8901234576, 'LEASE AGREEMENT-GA116', '2023-10-15', '2024-09-20', 2),

('Christopher Hernandez', 'christopherhernandez@gmail.com', 9012346678, 'LEASE AGREEMENT-GA117', '2023-07-25', '2024-08-30', 2),

('Emily Gonzalez', 'emilygonzalez@gmail.com', 1234567890, 'LEASE AGREEMENT-GA118', '2023-09-30', '2024-10-10', 2),

('Michael Martinez', 'michaelmartinez@gmail.com', 2356789140, 'LEASE AGREEMENT-GA119', '2023-08-05', '2024-12-05', 2),

('Emma Walker', 'emmawalker@gmail.com', 2345789016, 'LEASE AGREEMENT-OD101', '2023-10-20', '2024-09-10', 3),

('James Harris', 'jamesharris@gmail.com', 3467890125, 'LEASE AGREEMENT-OD102', '2023-08-05', '2024-12-15', 3),

('Olivia Robinson', 'oliviarobinson@gmail.com', 4578901236, 'LEASE AGREEMENT-OD103', '2023-06-25', '2024-11-20', 3),

('Lucas Young', 'lucasyoung@gmail.com', 5689012347, 'LEASE AGREEMENT-OD104', '2023-11-10', '2024-08-25', 3),

('Ava Martinez', 'avamartinez@gmail.com', 6790123458, 'LEASE AGREEMENT-OD105', '2023-07-15', '2024-10-05', 3),

('Daniel Hernandez', 'danielhernandez@gmail.com', 7801234569, 'LEASE AGREEMENT-OD106', '2023-05-30', '2024-09-30', 3),

('Sophia Diaz', 'sophiadiaz@gmail.com', 8901245673, 'LEASE AGREEMENT-OD107', '2023-10-05', '2024-12-10', 3),

('Jacob Rodriguez', 'jacobrodriguez@gmail.com', 9012456783, 'LEASE AGREEMENT-OD108', '2023-06-15', '2024-10-20', 3),

('Isabella Martinez', 'isabellamartinez@gmail.com', 1245678903, 'LEASE AGREEMENT-OD109', '2023-08-20', '2024-11-15', 3),

('Alexander Lopez', 'alexanderlopez@gmail.com', 2345689017, 'LEASE AGREEMENT-OD110', '2023-07-05', '2024-08-15', 3),

('Mia Gonzalez', 'miagonzalez@gmail.com', 3468012579, 'LEASE AGREEMENT-OD111', '2023-09-10', '2024-09-05', 3),

('Ethan Perez', 'ethanperez@gmail.com', 4567823901, 'LEASE AGREEMENT-OD112', '2023-06-30', '2024-12-01', 3),

('Sophia Smith', 'sophiasmith@gmail.com', 5791234680, 'LEASE AGREEMENT-OD113', '2023-11-05', '2024-08-10', 3),

('Jackson Anson', 'jacksonanson@gmail.com', 6781245390, 'LEASE AGREEMENT-OD114', '2023-08-10', '2024-10-15', 3),

('Charlotte Wilson', 'charlottewilson@gmail.com', 7893456201, 'LEASE AGREEMENT-OD115', '2023-05-20', '2024-11-30', 3),

('Logan Taylor', 'logantaylor@gmail.com', 8901246754, 'LEASE AGREEMENT-OD116', '2023-10-15', '2024-09-20', 3),

('Amelia White', 'ameliawhite@gmail.com', 9012357864, 'LEASE AGREEMENT-OD117', '2023-07-25', '2024-08-30', 3),

('Mason Jackson', 'masonjackson@gmail.com', 4567890321, 'LEASE AGREEMENT-OD118', '2023-09-30', '2024-10-10', 3),

('Ell Haris', 'ellharis@gmail.com', 2345678109, 'LEASE AGREEMENT-OD119', '2023-08-05', '2024-12-05', 3),

('Sophia Brown', 'sophiabrown@gmail.com', 2345679018, 'LEASE AGREEMENT-MA101', '2023-10-20', '2024-09-10', 4),

('William Davis', 'williamdavis@gmail.com', 3457890126, 'LEASE AGREEMENT-MA102', '2023-08-05', '2024-12-15', 4),

('Isabella Miller', 'isabellamiller@gmail.com', 4568901237, 'LEASE AGREEMENT-MA103', '2023-06-25', '2024-11-20', 4),

('Mason Wilson', 'masonwilson@gmail.com', 5679012348, 'LEASE AGREEMENT-MA104', '2023-11-10', '2024-08-25', 4),

('Emma Moore', 'emmamoore@gmail.com', 6780123459, 'LEASE AGREEMENT-MA105', '2023-07-15', '2024-10-05', 4),

('Olivia Taylor', 'oliviataylor@gmail.com', 7891234560, 'LEASE AGREEMENT-MA106', '2023-05-30', '2024-09-30', 4),

('Jackson Anderson', 'jacksonanderson@gmail.com', 8902345671, 'LEASE AGREEMENT-MA107', '2023-10-05', '2024-12-10', 4),

('Ava Thomas', 'avathomas@gmail.com', 9012346785, 'LEASE AGREEMENT-MA108', '2023-06-15', '2024-10-20', 4),

('James White', 'jameswhite@gmail.com', 1345678902, 'LEASE AGREEMENT-MA109', '2023-08-20', '2024-11-15', 4),

('Ella Harris', 'ellaharris@gmail.com', 2345678019, 'LEASE AGREEMENT-MA110', '2023-07-05', '2024-08-15', 4),

('Logan Clark', 'loganclark@gmail.com', 3589012467, 'LEASE AGREEMENT-MA111', '2023-09-10', '2024-09-05', 4),

('Amelia Lee', 'ameliale@gmail.com', 4678023591, 'LEASE AGREEMENT-MA112', '2023-06-30', '2024-12-01', 4),

('Michael Young', 'michaelyoung@gmail.com', 5689234601, 'LEASE AGREEMENT-MA113', '2023-11-05', '2024-08-10', 4),

('Sophia Fernandez', 'sophiafernandez@gmail.com', 6902345187, 'LEASE AGREEMENT-MA114', '2023-08-10', '2024-10-15', 4),

('Daniel Rodriguez', 'danielrodriguez@gmail.com', 7901346825, 'LEASE AGREEMENT-MA115', '2023-05-20', '2024-11-30', 4),

('Aiden Martinez', 'aidenmartinez@gmail.com', 8934567120, 'LEASE AGREEMENT-MA116', '2023-10-15', '2024-09-20', 4),

('Abigail King', 'abigailking@gmail.com', 9012578634, 'LEASE AGREEMENT-MA117', '2023-07-25', '2024-08-30', 4),

('Madison Perez', 'madisonperez@gmail.com', 1234780569, 'LEASE AGREEMENT-MA118', '2023-09-30', '2024-10-10', 4),

('Carter Rivera', 'carterrivera@gmail.com', 2456891037, 'LEASE AGREEMENT-MA119', '2023-08-05', '2024-12-05', 4),

('Charlotte Walker', 'charlottewalker@gmail.com', 2345678910, 'LEASE AGREEMENT-EE101', '2023-10-20', '2024-09-10', 5),

('William Harris', 'williamharris@gmail.com', 3456890127, 'LEASE AGREEMENT-EE102', '2023-08-05', '2024-12-15', 5),

('Sophia Robinson', 'sophiarobinson@gmail.com', 4567901238, 'LEASE AGREEMENT-EE103', '2023-06-25', '2024-11-20', 5),

('Logan Young', 'loganyoung@gmail.com', 5678012349, 'LEASE AGREEMENT-EE104', '2023-11-10', '2024-08-25', 5),

('Emma Martinez', 'emmamartinez@gmail.com', 6789123450, 'LEASE AGREEMENT-EE105', '2023-07-15', '2024-10-05', 5),

('Jacob Taylor', 'jacobtaylor@gmail.com', 7890234561, 'LEASE AGREEMENT-EE106', '2023-05-30', '2024-09-30', 5),

('Ava Diaz', 'avadiaz@gmail.com', 8901345672, 'LEASE AGREEMENT-EE107', '2023-10-05', '2024-12-10', 5),

('Michael Rodriguez', 'michaelrodriguez@gmail.com', 9012345687, 'LEASE AGREEMENT-EE108', '2023-06-15', '2024-10-20', 5),

('Isabell Marnez', 'isabellmarnez@gmail.com', 1234568907, 'LEASE AGREEMENT-EE109', '2023-08-20', '2024-11-15', 5),

('Alexander Lepoz', 'alexanderlepoz@gmail.com', 2347890156, 'LEASE AGREEMENT-EE110', '2023-07-05', '2024-08-15', 5),

('Sophia Gonzalez', 'sophiagonzalez@gmail.com', 3467901285, 'LEASE AGREEMENT-EE111', '2023-09-10', '2024-09-05', 5),

('James Perez', 'jamesperez@gmail.com', 4568902371, 'LEASE AGREEMENT-EE112', '2023-06-30', '2024-12-01', 5),

('Emma Smith', 'emmasmith@gmail.com', 5601234111, 'LEASE AGREEMENT-EE113', '2023-11-05', '2024-08-10', 5),

('Jacob Hernandez', 'jacobhernandez@gmail.com', 6789112345, 'LEASE AGREEMENT-EE114', '2023-08-10', '2024-10-15', 5),

('Olivia Rodriguez', 'oliviarodriguez@gmail.com', 7890223456, 'LEASE AGREEMENT-EE115', '2023-05-20', '2024-11-30', 5),

('Ethan Martinez', 'ethanmartinez@gmail.com', 8901334567, 'LEASE AGREEMENT-EE116', '2023-10-15', '2024-09-20', 5),

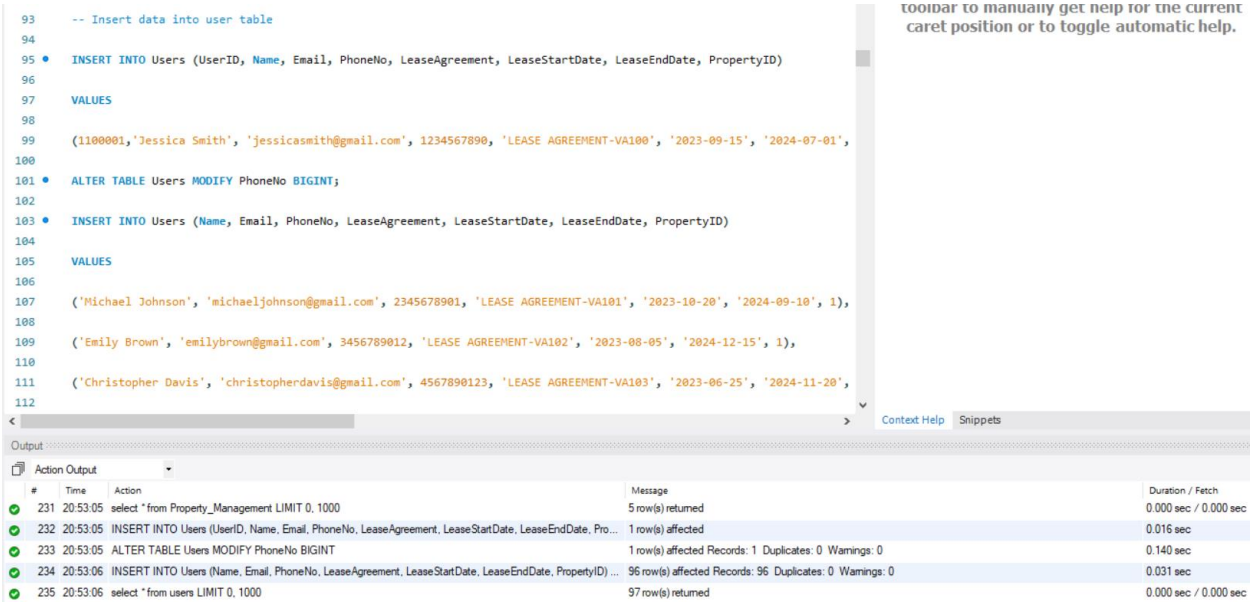
('Ava White', 'avawhite@gmail.com', 9013345678, 'LEASE AGREEMENT-EE117', '2023-07-25', '2024-08-30', 5),

('William Clark', 'williamclark@gmail.com', 1234667890, 'LEASE AGREEMENT-EE118', '2023-09-30', '2024-10-10', 5),

('Madison Lee', 'madisonlee@gmail.com', 2355678901, 'LEASE AGREEMENT-EE119', '2023-08-05', '2024-12-05', 5);

Above are the insert commands used to insert data into the Users entity. We made sure that each user's name is unique as the Name attribute has a constraint unique while creating the table Users.

Below is the screenshot of the Sql insert commands and respective output.



Using the below SQL command, we want to show the records of the Users Entity we just entered.

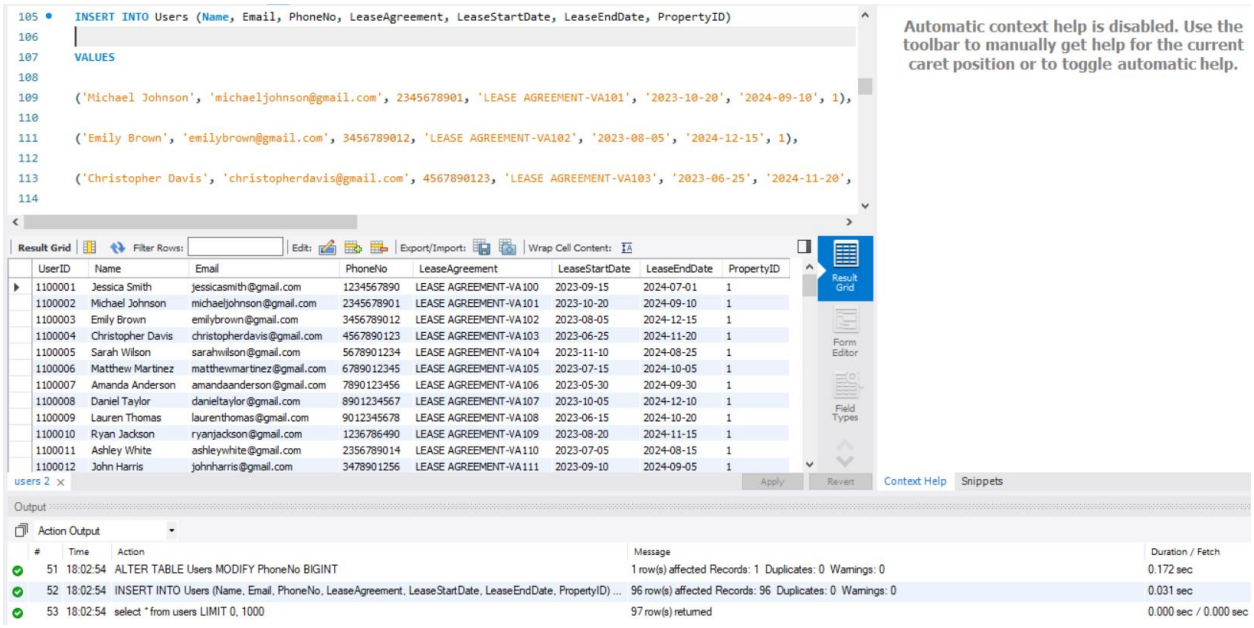
SQL Command:

Select * from Users;

Screenshot of command and its output:

```
283     ('James Perez', 'jamesperez@gmail.com', 4568902371, 'LEASE AGREEMENT-EE112', '2023-06-30', '2024-12-01', 5),
284
285     ('Emma Smith', 'emmasmith@gmail.com', 5601234111, 'LEASE AGREEMENT-EE113', '2023-11-05', '2024-08-10', 5),
286
287     ('Jacob Hernandez', 'jacobhernandez@gmail.com', 6789112345, 'LEASE AGREEMENT-EE114', '2023-08-10', '2024-10-15', 5),
288
289     ('Olivia Rodriguez', 'oliviarodriguez@gmail.com', 7890223456, 'LEASE AGREEMENT-EE115', '2023-05-20', '2024-11-30', 5)
290
291     ('Ethan Martinez', 'ethanmartinez@gmail.com', 8901334567, 'LEASE AGREEMENT-EE116', '2023-10-15', '2024-09-20', 5),
292
293     ('Ava White', 'avawhite@gmail.com', 9013345678, 'LEASE AGREEMENT-EE117', '2023-07-25', '2024-08-30', 5),
294
295     ('William Clark', 'williamclark@gmail.com', 1234667890, 'LEASE AGREEMENT-EE118', '2023-09-30', '2024-10-10', 5),
296
297     ('Madison Lee', 'madisonlee@gmail.com', 2355678901, 'LEASE AGREEMENT-EE119', '2023-08-05', '2024-12-05', 5);
298
299 •   select * from users;
300
```

Output:



3)Data Entry into **Category** entity:

By providing the attribute values and by using the insert command data is added to the Category entity. And for the first entry we assigned Category ID as 1401. There by the CategoryID for

following records will be incremented based upon the CategoryID of the first record as we know it is defined as an auto increment attribute.

Only 6 number of records were added to the Category entity. We thought like only the major issues or requests raised by users would fall into this category. Those categories include carpentry, landscaping, HVAC (Heating, Ventilation, Air Conditioning), Plumbing, Electrical, Appliance Repair. And according to the priorities we also ranked them as High, Medium, Low based on how urgent the issue needs to be resolved.

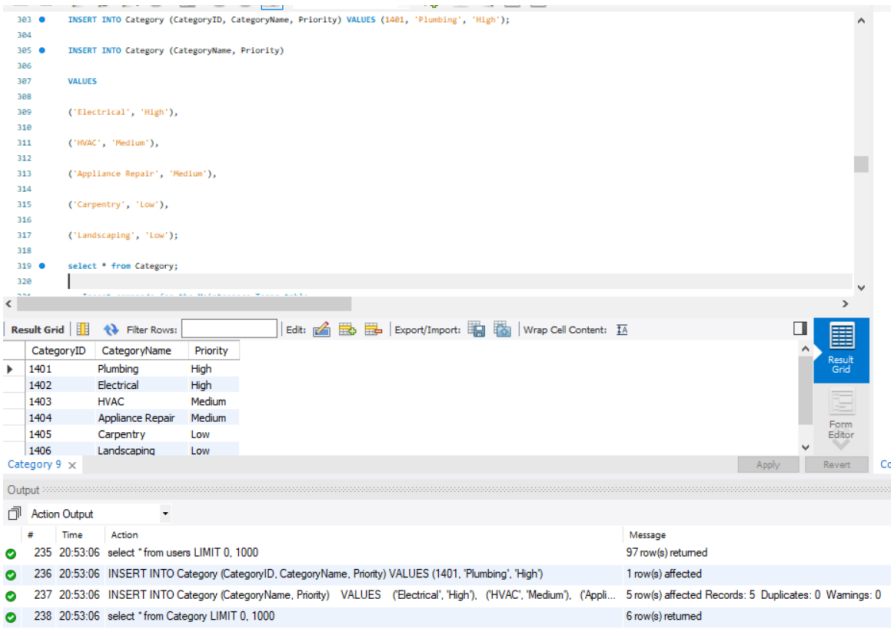
```
-- Insert commands for the Category table

INSERT INTO Category (CategoryID, CategoryName, Priority) VALUES (1401, 'Plumbing', 'High');

INSERT INTO Category (CategoryName, Priority)

VALUES

('Electrical', 'High'),

('HVAC', 'Medium'),

('Appliance Repair', 'Medium'),

('Carpentry', 'Low'),

('Landscaping', 'Low');
```

Using the below SQL command, we want to show the records of the Category Entity we just entered.

SQL Command:

```
Select * from Category;
```

Screenshot of insert and select commands and their output:



4)Data Entry into **Maintenance_Teams** entity:

We have entered data into the Maintenance_Teams entity using insert command by providing values for the attributes. We gave the TeamID for the first record only as 1501. As it is defined as auto increment attribute the TeamID for next records will be incremented based on the first record's TeamID.

We have inserted only 6 records into the Maintenance_Teams entity. As we assumed requests into only 6 categories in category entity. We have entered those teams which work in those categories. Which are the Plumbing Team, Electrical Team, HVAC (Heating, Ventilation and Air Conditioning) Team, Appliance Repair Team, Carpentry Team and Lanscaping Team. We gave the locations of those teams.

-- Insert commands for the Maintenance_Teams table

INSERT INTO Maintenance_Teams (TeamID, TeamName, Location) VALUES (1501, 'Plumbing Team', '123 Oak Street, Denton, TX');

INSERT INTO Maintenance_Teams (TeamName, Location)

VALUES

('Electrical Team', '456 Maple Avenue, Denton, TX'),

('HVAC Team', '789 Pine Road, Denton, TX'),

('Appliance Repair Team', '987 Cedar Street, Denton, TX'),

('Carpentry Team', '101 Elm Boulevard, Denton, TX'),

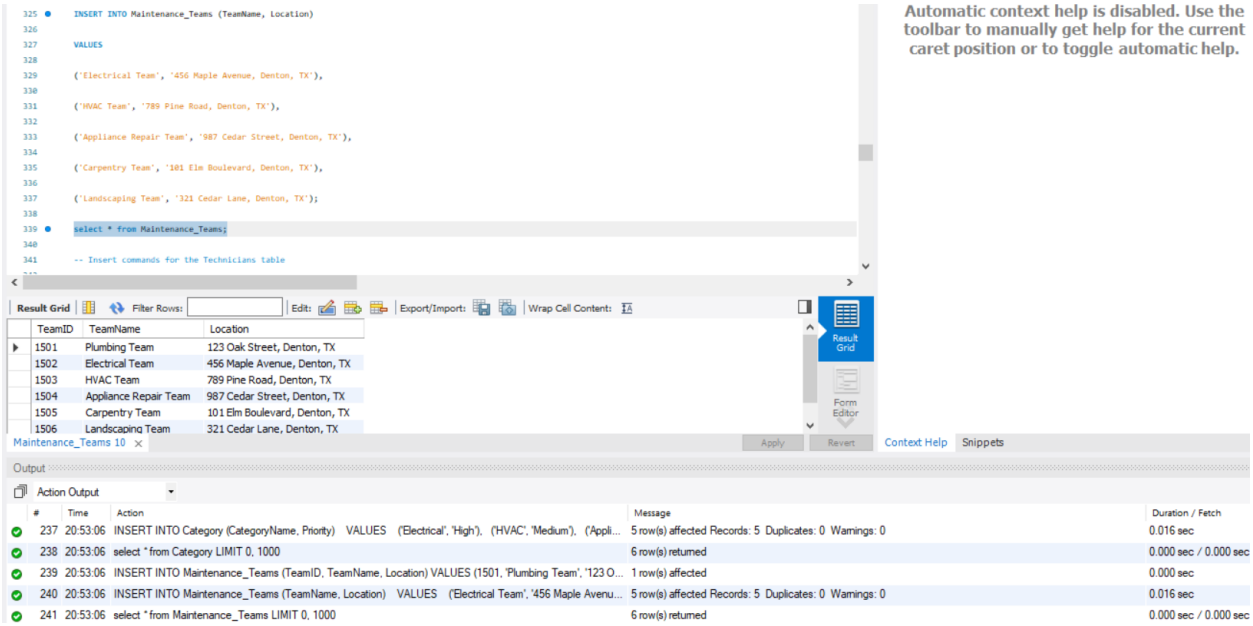
('Landscaping Team', '321 Cedar Lane, Denton, TX');

Using the below SQL command, we want to show the records of the Maintenance_Teams Entity we just entered.

SQL Command:

select * from Maintenance_Teams;

Screenshot of insert and select commands and their output:



5)Data Entry into **Technicians** entity:

With the help of Insert command, we have supplied data to the Technicians entity by providing values for the attributes. For the first record we have given Technician Id as 1501. Since it is defined as auto increment attribute, the Technician Id will be incremented having base as first record.

And then we inserted records into the Maintenance_Teams entity. And each technician's team ID their name along with their specialized area is given.

-- Insert commands for the Technicians table

```
INSERT INTO Technicians (TechnicianID, TeamID, Name, Role) VALUES
(1601,1501, 'John Smith', 'Plumber');
INSERT INTO Technicians (TeamID, Name, Role) VALUES
(1501, 'Emily Johnson', 'Plumber'),
(1501, 'Michael Williams', 'Plumber'),
(1501, 'Jessica Brown', 'Plumber'),
(1501, 'Christopher Davis', 'Plumber'),
(1502, 'David Jones', 'Electrician'),
(1502, 'Sarah Martinez', 'Electrician'),
(1502, 'Robert Miller', 'Electrician'),
(1502, 'Linda Wilson', 'Electrician'),
(1502, 'Daniel Taylor', 'Electrician'),
(1503, 'William Anderson', 'HVAC Technician'),
(1503, 'Jennifer Garcia', 'HVAC Technician'),
(1503, 'Thomas Lee', 'HVAC Technician'),
(1503, 'Margaret Rodriguez', 'HVAC Technician'),
(1503, 'Richard Hernandez', 'HVAC Technician'),
(1504, 'Jennifer Turner', 'Appliance Technician'),
(1504, 'Brian Scott', 'Appliance Technician'),
(1504, 'Lisa King', 'Appliance Technician'),
(1504, 'Anthony Perez', 'Appliance Technician'),
(1504, 'Karen Green', 'Appliance Technician'),
(1505, 'Mary Martinez', 'Carpenter'),
(1505, 'James Gonzalez', 'Carpenter'),
(1505, 'Patricia Wright', 'Carpenter'),
(1505, 'Charles Lopez', 'Carpenter'),
(1505, 'Angela Hill', 'Carpenter'),
(1506, 'Mark Young', 'Landscaper'),
(1506, 'Anna Moore', 'Landscaper'),
(1506, 'Joseph Clark', 'Landscaper'),
```

(1506, 'Elizabeth Lewis', 'Landscaper'),

(1506, 'Kevin Hall', 'Landscaper');

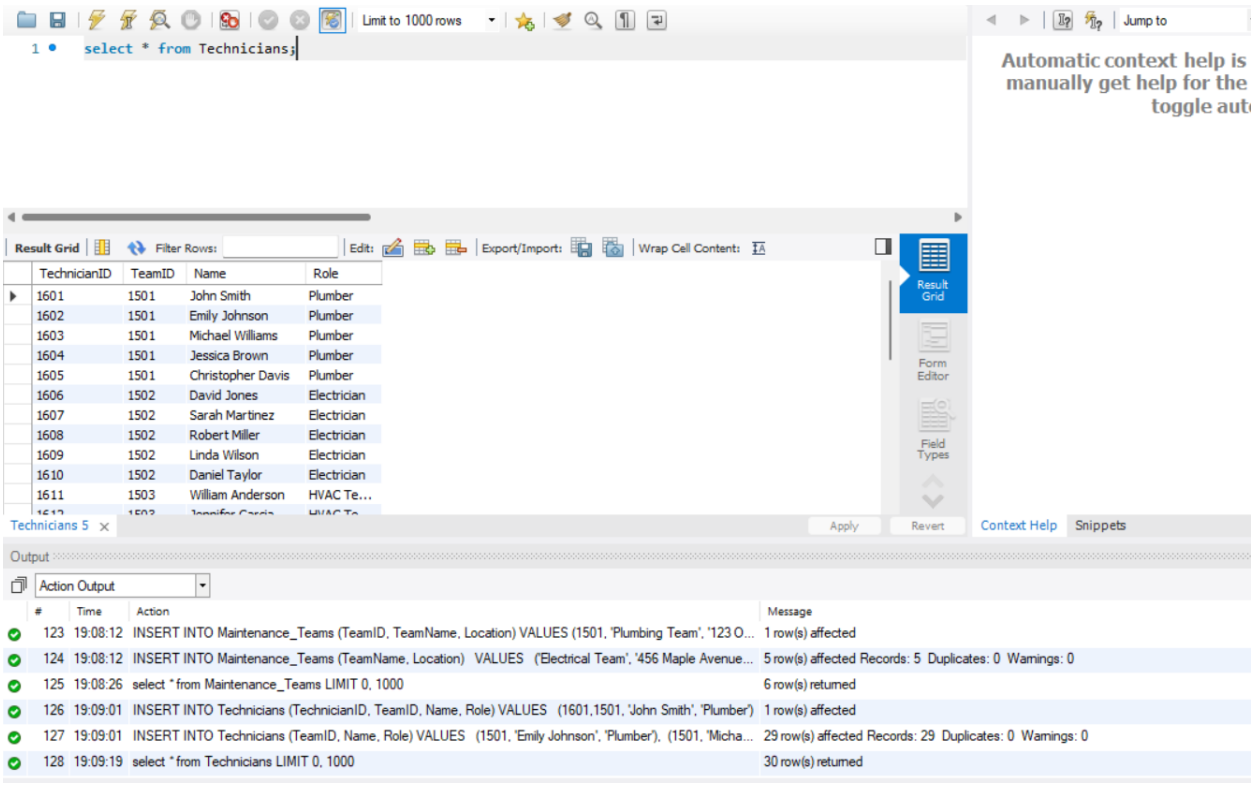
Using the below SQL command, we want to show the records of the Technicians Entity we just entered.

SQL Command:

Select * from Technicians;

Screenshot of insert and select commands and their output:

```
343 ●    INSERT INTO Technicians (TechnicianID, TeamID, Name, Role) VALUES
344
345      (1601,1501, 'John Smith', 'Plumber');
346
347 ●    INSERT INTO Technicians (TeamID, Name, Role) VALUES
348
349      (1501, 'Emily Johnson', 'Plumber'),
350
351      (1501, 'Michael Williams', 'Plumber'),
352
353      (1501, 'Jessica Brown', 'Plumber'),
354
355      (1501, 'Christopher Davis', 'Plumber'),
356
357      (1502, 'David Jones', 'Electrician'),
358
359      (1502, 'Sarah Martinez', 'Electrician'),
```



6)Data Entry into **Maintenance_Requests** entity:

We have entered data into the Maintenance_Requests entity using insert command by providing values for the attributes. We gave the RequestID for the first record starting with 1201.

We have inserted records into the Maintenance_Teams entity. We have given the UserID (the user who raised the request), CategoryID (request belongs to which category and its priority) for every request. Also gave Description, RequestRaisedDate and status (default we gave every request's status as Open).

```
-- Create a trigger to log entry request actions on Maintenance_Requests table

DELIMITER //

CREATE TRIGGER maintenance_request_insert_trigger

AFTER INSERT ON Maintenance_Requests

FOR EACH ROW

BEGIN

    -- Get the status of the updated request
```

```
    DECLARE new_status VARCHAR(20);
    SELECT Status INTO new_status FROM Maintenance_Requests WHERE RequestID = NEW.RequestID;

    -- Insert a record into Maintenance_Log for each action
    INSERT INTO Maintenance_Log (RequestID, Action, Timestamp)
    VALUES (NEW.RequestID, CONCAT('Status changed to: ', new_status), NOW());
END//

DELIMITER ;
```

-- Insert commands for the Maintenance_Requests table

INSERT INTO Maintenance_Requests (RequestID, UserID, CategoryID, TeamID, Location, Description, RequestRaisedDate, Status) VALUES

(1201, 1100001, 1401, 1501, '1408 Bernard St, Denton, 76201', 'Leaky faucet in the kitchen', '2024-04-27', 'Open'),

(1202, 1100003, 1402, 1502, '1408 Bernard St, Denton, 76201', 'Light fixture not working in the living room', '2024-04-27', 'Open'),

(1203, 1100004, 1403, 1503, '1408 Bernard St, Denton, 76201', 'AC unit not cooling properly', '2024-04-27', 'Open'),

(1204, 1100006, 1404, 1504, '1408 Bernard St, Denton, 76201', 'Door hinge loose in the bedroom', '2024-04-27', 'Open'),

(1205, 1100009, 1405, 1505, '1408 Bernard St, Denton, 76201', 'Grass needs cutting in the backyard', '2024-04-27', 'Open'),

(1206, 1100010, 1401, 1501, '1408 Bernard St, Denton, 76201', 'Sink clogged in the bathroom', '2024-04-27', 'Open'),

(1207, 1100018, 1402, 1502, '1408 Bernard St, Denton, 76201', 'Outlet not working in the kitchen', '2024-04-27', 'Open'),

(1208, 1100022, 1403, 1503, 'South Carroll Blvd st Denton, 76201', 'AC unit making strange noise', '2024-04-27', 'Open'),

(1209, 1100027, 1404, 1504, 'South Carroll Blvd st Denton, 76201', 'Drawer stuck in the living room', '2024-04-27', 'Open'),

(1210, 1100023, 1405, 1505, 'South Carroll Blvd st Denton, 76201', 'Shrubbery needs trimming in the front yard', '2024-04-27', 'Open'),

(1211, 1100025, 1401, 1501, 'South Carroll Blvd st Denton, 76201', 'Toilet running in the bathroom', '2024-04-27', 'Open'),

(1212, 1100029, 1402, 1502, 'South Carroll Blvd st Denton, 76201', 'Ceiling fan not turning on in the bedroom', '2024-04-27', 'Open'),

(1213, 1100031, 1403, 1503, 'South Carroll Blvd st Denton, 76201', 'Heater not heating properly in the living room', '2024-04-27', 'Open'),

(1214, 1100033, 1404, 1504, 'South Carroll Blvd st Denton, 76201', 'Closet door off its track in the hallway', '2024-04-27', 'Open'),

(1215, 1100038, 1405, 1505, 'South Carroll Blvd st Denton, 76201', 'Garden bed needs weeding in the backyard', '2024-04-27', 'Open'),

(1216, 1100037, 1401, 1501, 'South Carroll Blvd st Denton, 76201', 'Shower head leaking in the bathroom', '2024-04-27', 'Open'),

(1217, 1100041, 1402, 1502, '425 Bernard St, Denton, TX 76201', 'Light flickering in the dining room', '2024-04-27', 'Open'),

(1218, 1100044, 1403, 1503, '425 Bernard St, Denton, TX 76201', 'AC unit not turning on in the office', '2024-04-27', 'Open'),

(1219, 1100047, 1404, 1504, '425 Bernard St, Denton, TX 76201', 'Cabinet door broken in the kitchen', '2024-04-27', 'Open'),

(1220, 1100052, 1405, 1505, '425 Bernard St, Denton, TX 76201', 'Mulch needs replenishing in the front yard', '2024-04-27', 'Open'),

(1221, 1100053, 1401, 1501, '425 Bernard St, Denton, TX 76201', 'Garbage disposal jammed in the kitchen', '2024-04-27', 'Open'),

(1222, 1100056, 1402, 1502, '425 Bernard St, Denton, TX 76201', 'Bathroom exhaust fan not working in the bathroom', '2024-04-27', 'Open'),

(1223, 1100055, 1403, 1503, '425 Bernard St, Denton, TX 76201', 'Thermostat not responding in the living room', '2024-04-27', 'Open'),

(1224, 1100066, 1404, 1504, '627 Bernard St, Denton, TX 76201', 'Drawer handle loose in the bedroom', '2024-04-27', 'Open'),

(1225, 1100067, 1405, 1505, '627 Bernard St, Denton, TX 76201', 'Lawn needs mowing in the backyard', '2024-04-27', 'Open'),

(1226, 1100069, 1401, 1501, '627 Bernard St, Denton, TX 76201', 'Leak under the sink in the kitchen', '2024-04-27', 'Open'),

(1227, 1100070, 1402, 1502, '627 Bernard St, Denton, TX 76201', 'Outlet sparking in the living room', '2024-04-27', 'Open'),

(1228, 1100071, 1403, 1503, '627 Bernard St, Denton, TX 76201', 'AC unit leaking water in the bedroom', '2024-04-27', 'Open'),

(1229, 1100075, 1404, 1504, '627 Bernard St, Denton, TX 76201', 'Closet door stuck in the hallway', '2024-04-27', 'Open'),

(1230, 1100072, 1405, 1505, '627 Bernard St, Denton, TX 76201', 'Tree limb needs trimming in the backyard', '2024-04-27', 'Open'),

(1231, 1100073, 1401, 1501, '627 Bernard St, Denton, TX 76201', 'Toilet not flushing in the bathroom', '2024-04-27', 'Open'),

(1232, 1100081, 1402, 1502, '903 Avenue C, Denton, TX 76201', 'Ceiling light not turning off in the bedroom', '2024-04-27', 'Open'),

(1233, 1100061, 1403, 1503, '627 Bernard St, Denton, TX 76201', 'Heater blowing cold air in the living room', '2024-04-27', 'Open'),

(1234, 1100031, 1404, 1504, 'South Carroll Blvd st Denton, 76201', 'Drawer wont close in the kitchen', '2024-04-27', 'Open'),

(1235, 1100021, 1405, 1505, 'South Carroll Blvd st Denton, 76201', 'Bushes need trimming in the front yard', '2024-04-27', 'Open'),

(1236, 1100011, 1401, 1501, '1408 Bernard St, Denton, 76201', 'Sink faucet dripping in the bathroom', '2024-04-27', 'Open'),

(1237, 1100051, 1402, 1502, '425 Bernard St, Denton, TX 76201', 'Dimmer switch not working in the dining room', '2024-04-27', 'Open'),

(1238, 1100088, 1403, 1503, '903 Avenue C, Denton, TX 76201', 'AC unit blowing warm air in the office', '2024-04-27', 'Open'),

(1239, 1100083, 1404, 1504, '903 Avenue C, Denton, TX 76201', 'Shelf bracket broken in the kitchen', '2024-04-27', 'Open'),

(1240, 1100084, 1405, 1505, '903 Avenue C, Denton, TX 76201', 'Weeds need pulling in the backyard', '2024-04-27', 'Open'),

(1241, 1100086, 1406, 1506, '903 Avenue C, Denton, TX 76201', 'Refrigerator not cooling properly', '2024-04-27', 'Open'),

(1242, 1100054, 1406, 1506, '425 Bernard St, Denton, TX 76201', 'Dishwasher not draining', '2024-04-27', 'Open'),

(1243, 1100050, 1406, 1506, '425 Bernard St, Denton, TX 76201', 'Oven not heating', '2024-04-27', 'Open'),

(1244, 1100068, 1406, 1506, '627 Bernard St, Denton, TX 76201', 'Washing machine not spinning', '2024-04-27', 'Open'),

(1245, 1100073, 1406, 1506, '627 Bernard St, Denton, TX 76201', 'Dryer not drying clothes', '2024-04-27', 'Open');

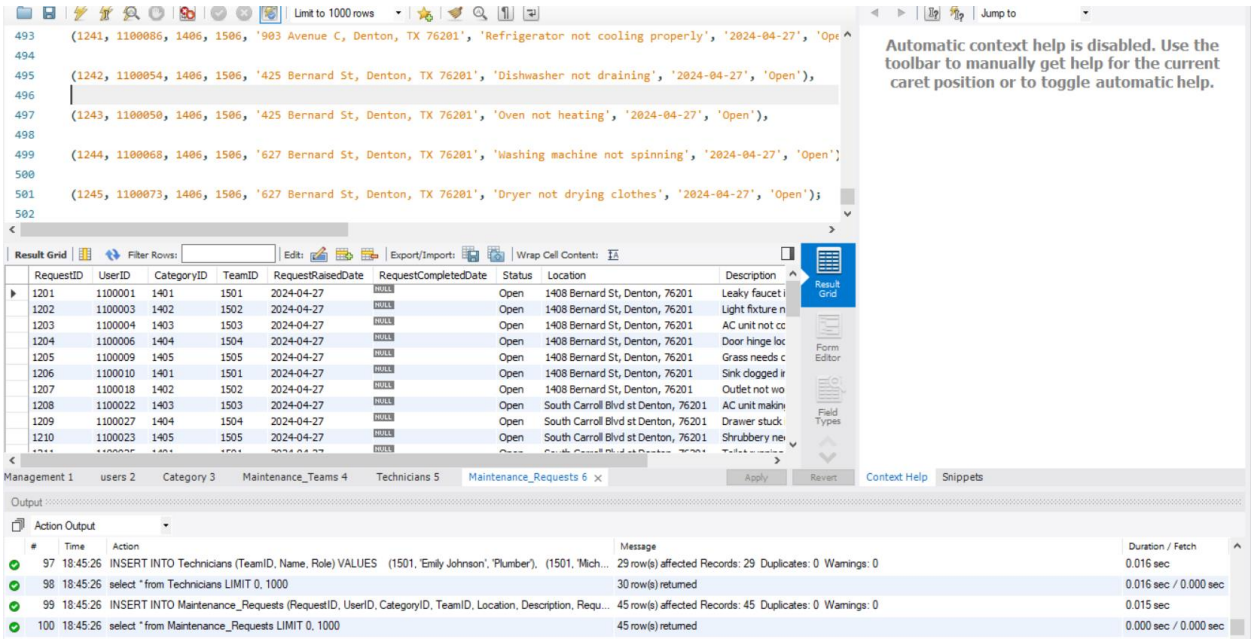
Using the below SQL command, we want to show the records of the Maintenance_Requests Entity we just entered.

SQL Command:

Select * from Maintenance_Requests;

Screenshot of insert and select commands and their output:

```
428 ●   -- Insert commands for the Maintenance_Requests table
429
430     INSERT INTO Maintenance_Requests (RequestID, UserID, CategoryID, TeamID, Location, Description, RequestRaisedDate, Status) VALUES
431
432     (1201, 1100001, 1401, 1501, '1408 Bernard St, Denton, 76201', 'Leaky faucet in the kitchen', '2024-04-27', 'Open'),
433
434     (1202, 1100003, 1402, 1502, '1408 Bernard St, Denton, 76201', 'Light fixture not working in the living room', '2024-04-27', 'Open'),
435
436     (1203, 1100004, 1403, 1503, '1408 Bernard St, Denton, 76201', 'AC unit not cooling properly', '2024-04-27', 'Open'),
437
438     (1204, 1100006, 1404, 1504, '1408 Bernard St, Denton, 76201', 'Door hinge loose in the bedroom', '2024-04-27', 'Open'),
439
440     (1205, 1100009, 1405, 1505, '1408 Bernard St, Denton, 76201', 'Grass needs cutting in the backyard', '2024-04-27', 'Open'),
441
442     (1206, 1100010, 1401, 1501, '1408 Bernard St, Denton, 76201', 'Sink clogged in the bathroom', '2024-04-27', 'Open'),
443
444     (1207, 1100018, 1402, 1502, '1408 Bernard St, Denton, 76201', 'Outlet not working in the kitchen', '2024-04-27', 'Open'),
445
446     (1208, 1100022, 1403, 1503, 'South Carroll Blvd st Denton, 76201', 'AC unit making strange noise', '2024-04-27', 'Open'),
447
448     (1209, 1100027, 1404, 1504, 'South Carroll Blvd st Denton, 76201', 'Drawer stuck in the living room', '2024-04-27', 'Open'),
449
450     (1210, 1100023, 1405, 1505, 'South Carroll Blvd st Denton, 76201', 'Shrubbery needs trimming in the front yard', '2024-04-27', 'Open'),
```



Default values of RequestCompletedDate are NULL as the status is open for now. When we update the status to Completed, we give the date in RequestCompletedDate attribute.

```
-- Create a trigger to log update request info actions on Maintenance_Requests table
DELIMITER //
CREATE TRIGGER maintenance_request_update_trigger
AFTER UPDATE ON Maintenance_Requests
FOR EACH ROW
BEGIN
    -- Get the status of the updated request
    DECLARE new_status VARCHAR(20);
    SELECT Status INTO new_status FROM Maintenance_Requests WHERE RequestID = NEW.RequestID;

    -- Insert a record into Maintenance_Log for each action
    INSERT INTO Maintenance_Log (RequestID, Action, Timestamp)
    VALUES (NEW.RequestID, CONCAT('Status changed to: ', new_status), NOW());
END//

DELIMITER ;
```

7)Data Entry into **Maintenance_Log** entity:

In order to update the Maintenance_Log whenever a request id is inserted/updated in the Maintenance_Requests table, we have created triggers called maintenance_request_insert_trigger and maintenance_request_update_trigger. After that, we updated the Maintenance_Log table with request id, Action, timestamp when a request is inserted/updated.

Logs trigger:

```
409     -- Create a trigger to log entry request actions on Maintenance_Requests table
410
411     DELIMITER //
412
413 •   CREATE TRIGGER maintenance_request_insert_trigger
414     AFTER INSERT ON Maintenance_Requests
415     FOR EACH ROW
416     BEGIN
417         -- Get the status of the updated request
418         DECLARE new_status VARCHAR(20);
419         SELECT Status INTO new_status FROM Maintenance_Requests WHERE RequestID = NEW.RequestID;
420
421         -- Insert a record into Maintenance_Log for each action
422         INSERT INTO Maintenance_Log (RequestID, Action, Timestamp)
423         VALUES (NEW.RequestID, CONCAT('Status changed to: ', new_status), NOW());
424     END//
425
426     DELIMITER ;
```

```
524     -- Create a trigger to log update request info actions on Maintenance_Requests table
525
526     DELIMITER //
527
528 •   CREATE TRIGGER maintenance_request_update_trigger
529     AFTER UPDATE ON Maintenance_Requests
530     FOR EACH ROW
531     BEGIN
532         -- Get the status of the updated request
533         DECLARE new_status VARCHAR(20);
534         SELECT Status INTO new_status FROM Maintenance_Requests WHERE RequestID = NEW.RequestID;
535
536         -- Insert a record into Maintenance_Log for each action
537         INSERT INTO Maintenance_Log (RequestID, Action, Timestamp)
538         VALUES (NEW.RequestID, CONCAT('Status changed to: ', new_status), NOW());
539     END//
540
541     DELIMITER ;
```

These triggers are implemented in the creation process of the entities.

Now, we have updated a few requests' status to "In Progress" using the update commands for their respective request using RequestIDs with WHERE clause.

```
UPDATE Maintenance_Requests SET Status = 'In Progress' WHERE RequestID = 1201;

UPDATE Maintenance_Requests SET Status = 'In Progress' WHERE RequestID = 1202;
```

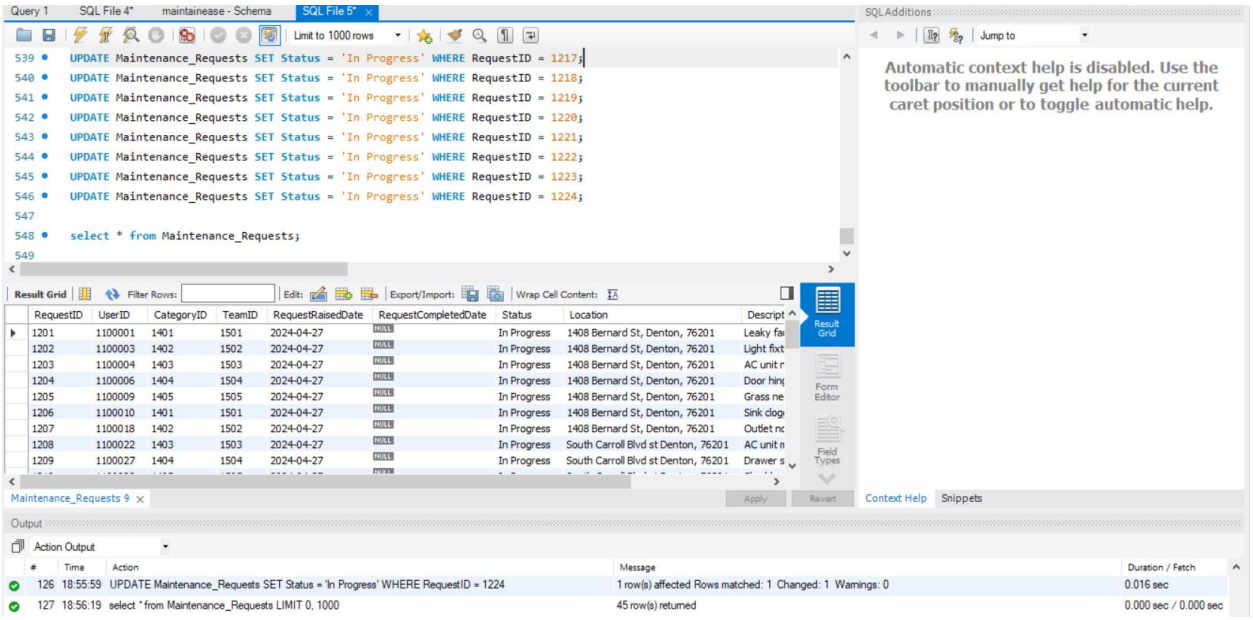
```
UPDATE Maintenance_Requests SET Status = 'In Progress' WHERE RequestID = 1203;
UPDATE Maintenance_Requests SET Status = 'In Progress' WHERE RequestID = 1204;
UPDATE Maintenance_Requests SET Status = 'In Progress' WHERE RequestID = 1205;
UPDATE Maintenance_Requests SET Status = 'In Progress' WHERE RequestID = 1206;
UPDATE Maintenance_Requests SET Status = 'In Progress' WHERE RequestID = 1207;
UPDATE Maintenance_Requests SET Status = 'In Progress' WHERE RequestID = 1208;
UPDATE Maintenance_Requests SET Status = 'In Progress' WHERE RequestID = 1209;
UPDATE Maintenance_Requests SET Status = 'In Progress' WHERE RequestID = 1210;
UPDATE Maintenance_Requests SET Status = 'In Progress' WHERE RequestID = 1211;
UPDATE Maintenance_Requests SET Status = 'In Progress' WHERE RequestID = 1212;
UPDATE Maintenance_Requests SET Status = 'In Progress' WHERE RequestID = 1213;
UPDATE Maintenance_Requests SET Status = 'In Progress' WHERE RequestID = 1214;
UPDATE Maintenance_Requests SET Status = 'In Progress' WHERE RequestID = 1215;
UPDATE Maintenance_Requests SET Status = 'In Progress' WHERE RequestID = 1216;
UPDATE Maintenance_Requests SET Status = 'In Progress' WHERE RequestID = 1217;
UPDATE Maintenance_Requests SET Status = 'In Progress' WHERE RequestID = 1218;
UPDATE Maintenance_Requests SET Status = 'In Progress' WHERE RequestID = 1219;
UPDATE Maintenance_Requests SET Status = 'In Progress' WHERE RequestID = 1220;
UPDATE Maintenance_Requests SET Status = 'In Progress' WHERE RequestID = 1221;
UPDATE Maintenance_Requests SET Status = 'In Progress' WHERE RequestID = 1222;
UPDATE Maintenance_Requests SET Status = 'In Progress' WHERE RequestID = 1223;
UPDATE Maintenance_Requests SET Status = 'In Progress' WHERE RequestID = 1224;
```

Using the below SQL command, we want to show the records updated of the Maintenance_Requests Entity we just entered.

SQL Command:

Select * from Maintenance_Requests;

Screenshot of update and select commands and their output:



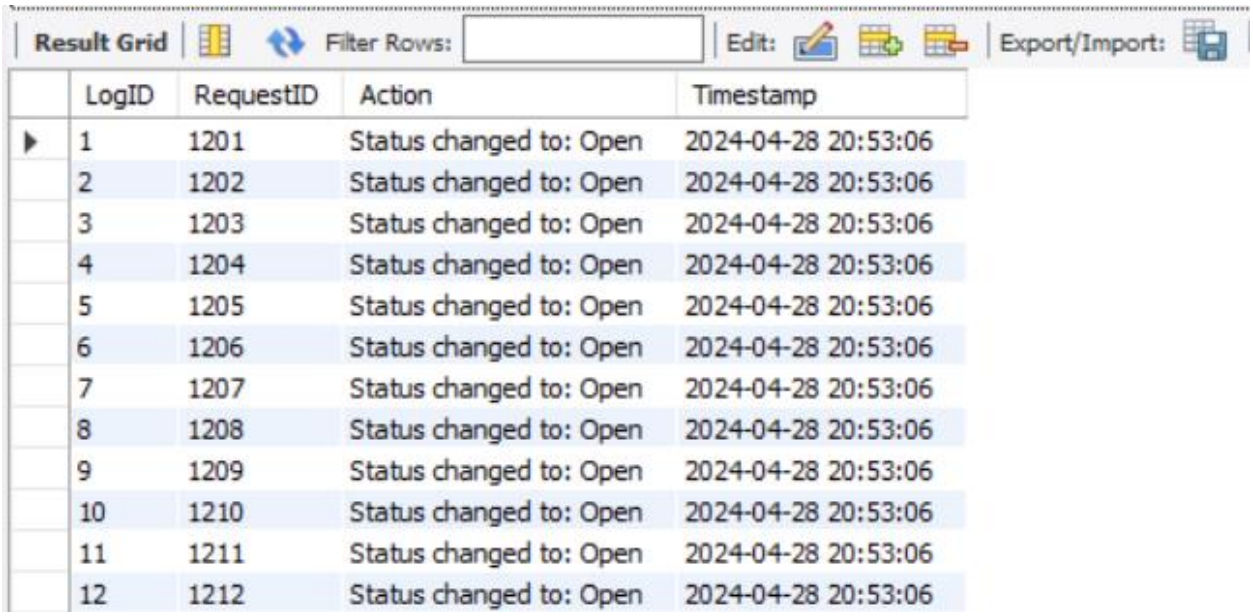
Logs triggered by update command on request in Maintenance_Requests entity:

Using the below SQL command, we want to show the records updated of the Maintenance_Logs Entity we just entered.

SQL Command:

Select * from Maintenance_Log;

Output:



LogID	RequestID	Action	Timestamp
1	1201	Status changed to: Open	2024-04-28 20:53:06
2	1202	Status changed to: Open	2024-04-28 20:53:06
3	1203	Status changed to: Open	2024-04-28 20:53:06
4	1204	Status changed to: Open	2024-04-28 20:53:06
5	1205	Status changed to: Open	2024-04-28 20:53:06
6	1206	Status changed to: Open	2024-04-28 20:53:06
7	1207	Status changed to: Open	2024-04-28 20:53:06
8	1208	Status changed to: Open	2024-04-28 20:53:06
9	1209	Status changed to: Open	2024-04-28 20:53:06
10	1210	Status changed to: Open	2024-04-28 20:53:06
11	1211	Status changed to: Open	2024-04-28 20:53:06
12	1212	Status changed to: Open	2024-04-28 20:53:06

LogID	RequestID	Action	Timestamp
42	1242	Status changed to: Open	2024-04-28 20:53:06
43	1243	Status changed to: Open	2024-04-28 20:53:06
44	1244	Status changed to: Open	2024-04-28 20:53:06
45	1245	Status changed to: Open	2024-04-28 20:53:06
46	1201	Status changed to: In Pr...	2024-04-28 20:53:06
47	1202	Status changed to: In Pr...	2024-04-28 20:53:06
48	1203	Status changed to: In Pr...	2024-04-28 20:53:06
49	1204	Status changed to: In Pr...	2024-04-28 20:53:06
50	1205	Status changed to: In Pr...	2024-04-28 20:53:06
51	1206	Status changed to: In Pr...	2024-04-28 20:53:06
52	1207	Status changed to: In Pr...	2024-04-28 20:53:06
53	1208	Status changed to: In Pr...	2024-04-28 20:53:06
54	1209	Status changed to: In Pr...	2024-04-28 20:53:06
55	1210	Status changed to: In Pr...	2024-04-28 20:53:06
56	1211	Status changed to: In Pr...	2024-04-28 20:53:06
57	1212	Status changed to: In Pr...	2024-04-28 20:53:06
58	1213	Status changed to: In Pr...	2024-04-28 20:53:06

We have changed the status of a few requests which are in progress state to completed using the below update commands.

```
UPDATE Maintenance_Requests SET Status = 'Completed' , RequestCompletedDate = '2024-04-28' WHERE RequestID = 1201;

UPDATE Maintenance_Requests SET Status = 'Completed' , RequestCompletedDate = '2024-04-28' WHERE RequestID = 1202;

UPDATE Maintenance_Requests SET Status = 'Completed' , RequestCompletedDate = '2024-04-28' WHERE RequestID = 1203;

UPDATE Maintenance_Requests SET Status = 'Completed' , RequestCompletedDate = '2024-04-28' WHERE RequestID = 1204;

UPDATE Maintenance_Requests SET Status = 'Completed' , RequestCompletedDate = '2024-04-28' WHERE RequestID = 1205;
```

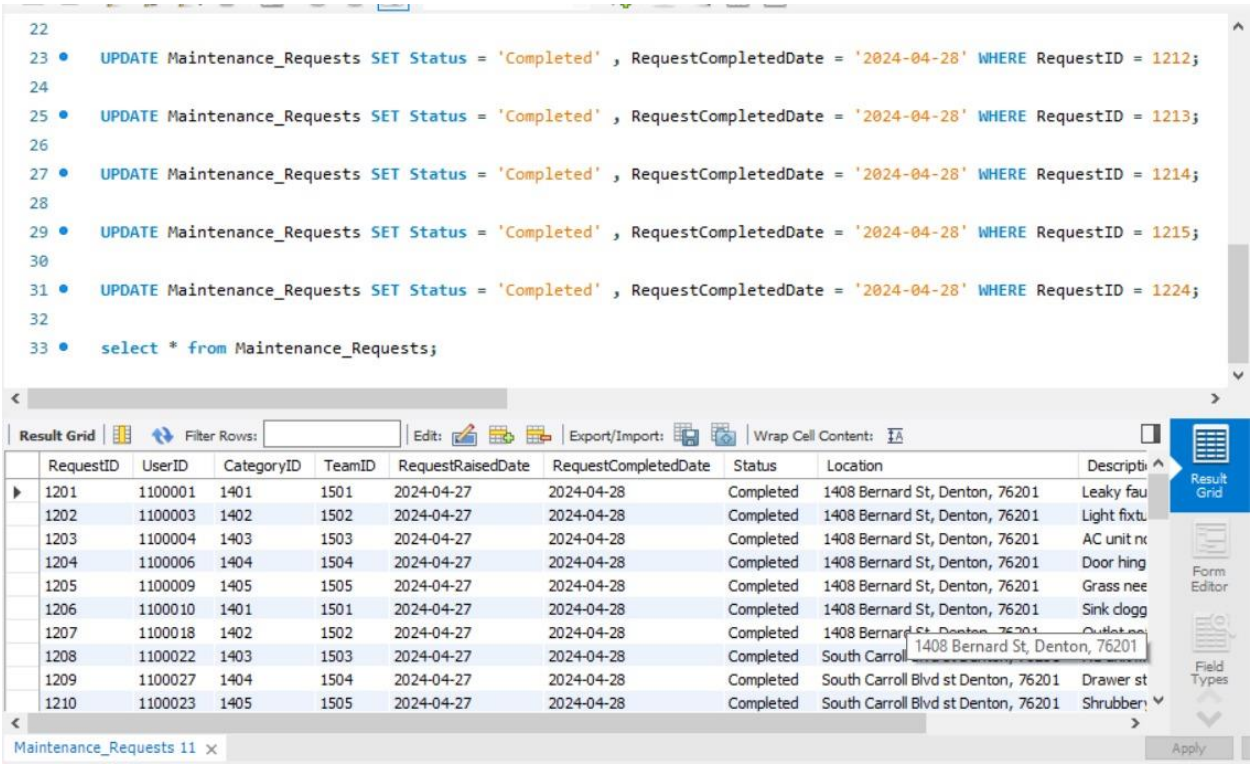
```
UPDATE Maintenance_Requests SET Status = 'Completed' , RequestCompletedDate = '2024-04-28' WHERE
RequestID = 1206;

UPDATE Maintenance_Requests SET Status = 'Completed' , RequestCompletedDate = '2024-04-28' WHERE
RequestID = 1207;

UPDATE Maintenance_Requests SET Status = 'Completed' , RequestCompletedDate = '2024-04-28' WHERE
RequestID = 1208;

UPDATE Maintenance_Requests SET Status = 'Completed' , RequestCompletedDate = '2024-04-28' WHERE
RequestID = 1209;

UPDATE Maintenance_Requests SET Status = 'Completed' , RequestCompletedDate = '2024-04-28' WHERE
RequestID = 1210;

UPDATE Maintenance_Requests SET Status = 'Completed' , RequestCompletedDate = '2024-04-28' WHERE
RequestID = 1211;

UPDATE Maintenance_Requests SET Status = 'Completed' , RequestCompletedDate = '2024-04-28' WHERE
RequestID = 1212;

UPDATE Maintenance_Requests SET Status = 'Completed' , RequestCompletedDate = '2024-04-28' WHERE
RequestID = 1213;

UPDATE Maintenance_Requests SET Status = 'Completed' , RequestCompletedDate = '2024-04-28' WHERE
RequestID = 1214;

UPDATE Maintenance_Requests SET Status = 'Completed' , RequestCompletedDate = '2024-04-28' WHERE
RequestID = 1215;

UPDATE Maintenance_Requests SET Status = 'Completed' , RequestCompletedDate = '2024-04-28' WHERE
RequestID = 1224;
```

Using the below SQL command, we want to show the records updated of the Maintenance_Requests Entity we just entered.

SQL Command:

```
Select * from Maintenance_Requests;
```

Screenshot of update and select commands and their output:



```
22
23 •  UPDATE Maintenance_Requests SET Status = 'Completed' , RequestCompletedDate = '2024-04-28' WHERE RequestID = 1212;
24
25 •  UPDATE Maintenance_Requests SET Status = 'Completed' , RequestCompletedDate = '2024-04-28' WHERE RequestID = 1213;
26
27 •  UPDATE Maintenance_Requests SET Status = 'Completed' , RequestCompletedDate = '2024-04-28' WHERE RequestID = 1214;
28
29 •  UPDATE Maintenance_Requests SET Status = 'Completed' , RequestCompletedDate = '2024-04-28' WHERE RequestID = 1215;
30
31 •  UPDATE Maintenance_Requests SET Status = 'Completed' , RequestCompletedDate = '2024-04-28' WHERE RequestID = 1224;
32
33 •  select * from Maintenance_Requests;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

RequestID	UserID	CategoryID	TeamID	RequestRaisedDate	RequestCompletedDate	Status	Location	Descripti
1201	1100001	1401	1501	2024-04-27	2024-04-28	Completed	1408 Bernard St, Denton, 76201	Leaky fau
1202	1100003	1402	1502	2024-04-27	2024-04-28	Completed	1408 Bernard St, Denton, 76201	Light fixtu
1203	1100004	1403	1503	2024-04-27	2024-04-28	Completed	1408 Bernard St, Denton, 76201	AC unit nc
1204	1100006	1404	1504	2024-04-27	2024-04-28	Completed	1408 Bernard St, Denton, 76201	Door hing
1205	1100009	1405	1505	2024-04-27	2024-04-28	Completed	1408 Bernard St, Denton, 76201	Grass nee
1206	1100010	1401	1501	2024-04-27	2024-04-28	Completed	1408 Bernard St, Denton, 76201	Sink clogg
1207	1100018	1402	1502	2024-04-27	2024-04-28	Completed	1408 Bernard St, Denton, 76201	Outlet no
1208	1100022	1403	1503	2024-04-27	2024-04-28	Completed	South Carroll	
1209	1100027	1404	1504	2024-04-27	2024-04-28	Completed	South Carroll Blvd st Denton, 76201	Drawer st
1210	1100023	1405	1505	2024-04-27	2024-04-28	Completed	South Carroll Blvd st Denton, 76201	Shrubber

1408 Bernard St, Denton, 76201

Result Grid | Form Editor | Field Types | Apply

Maintenance_Requests 11 ×

Logs triggered by update command on request in Maintenance_Requests entity:

Using the below SQL command, we want to show the records updated of the Maintenance_Logs Entity we just entered.

SQL Command:

Select * from Maintenance_Log;

Output:

LogID	RequestID	Action	Timestamp
67	1222	Status changed to: In Progress	2024-04-28 20:53:06
68	1223	Status changed to: In Progress	2024-04-28 20:53:06
69	1224	Status changed to: In Progress	2024-04-28 20:53:06
70	1201	Status changed to: Completed	2024-04-28 21:13:10
71	1202	Status changed to: Completed	2024-04-28 21:13:10
72	1203	Status changed to: Completed	2024-04-28 21:13:10
73	1204	Status changed to: Completed	2024-04-28 21:13:10
74	1205	Status changed to: Completed	2024-04-28 21:13:10
75	1206	Status changed to: Completed	2024-04-28 21:13:10
76	1207	Status changed to: Completed	2024-04-28 21:13:10
77	1208	Status changed to: Completed	2024-04-28 21:13:10
78	1209	Status changed to: Completed	2024-04-28 21:13:10
79	1210	Status changed to: Completed	2024-04-28 21:13:10
80	1211	Status changed to: Completed	2024-04-28 21:13:10
81	1212	Status changed to: Completed	2024-04-28 21:13:10
82	1213	Status changed to: Completed	2024-04-28 21:13:10
83	1214	Status changed to: Completed	2024-04-28 21:13:10
84	1215	Status changed to: Completed	2024-04-28 21:13:10
NULL	NULL	NULL	NULL

Section2: Data Retrieval and Simple Reports:

Data analysis 1:

To know the total number of requests from each property based on each category.

SQL Explanation:

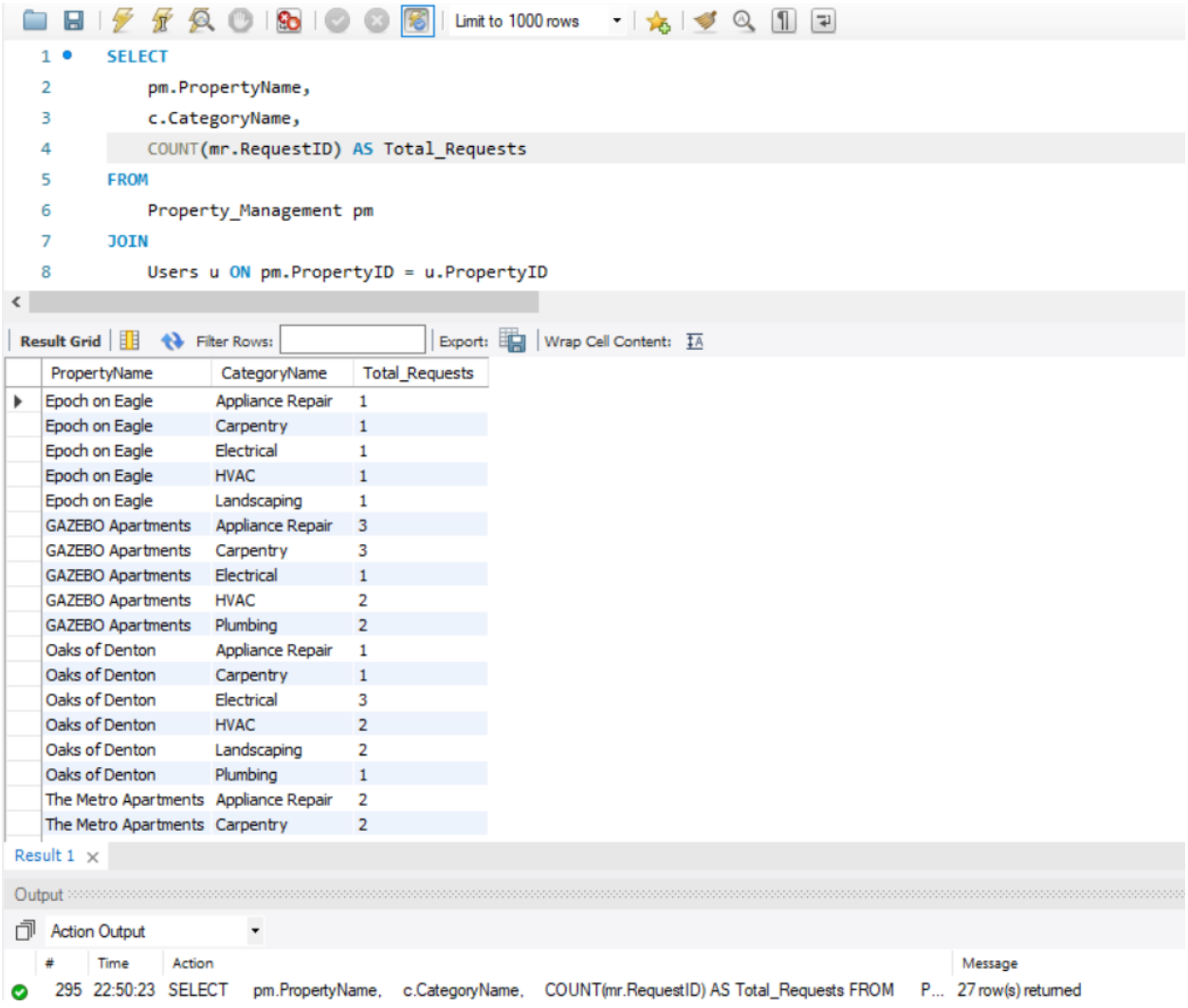
- Selects the PropertyName, CategoryName and Count() (counts the total number of requests for each property based on category) from the join functions.
- Joins the Property_Management as pm, Users as u, Maintenance_Requests as mr and Category as c.
- Joined the table Users with Property_Management table on PropertyID.
- Joined the table Maintenance_Requests with Users table on UsersID.
- Joined the table Maintenance_Requests with Category table on CategoryID.
- GROUP BY groups the results on PropertyName and CategoryName.
- ORDER BY orders the results on PropertyName and CategoryName.

SQL Query:

```
SELECT
   pm.PropertyName,
   c.CategoryName,
   COUNT(mr.RequestID) AS Total_Requests
FROM
   Property_Management pm
JOIN
   Users u ON pm.PropertyID = u.PropertyID
JOIN
   Maintenance_Requests mr ON u.UserID = mr.UserID
JOIN
   Category c ON mr.CategoryID = c.CategoryID
GROUP BY
   pm.PropertyName, c.CategoryName
ORDER BY
```

pm.PropertyName, c.CategoryName;

Output:



```
SELECT
    pm.PropertyName,
    c.CategoryName,
    COUNT(mr.RequestID) AS Total_Requests
FROM
    Property_Management pm
JOIN
    Users u ON pm.PropertyID = u.PropertyID
```

PropertyName	CategoryName	Total_Requests
Epoch on Eagle	Appliance Repair	1
Epoch on Eagle	Carpentry	1
Epoch on Eagle	Electrical	1
Epoch on Eagle	HVAC	1
Epoch on Eagle	Landscaping	1
GAZEBO Apartments	Appliance Repair	3
GAZEBO Apartments	Carpentry	3
GAZEBO Apartments	Electrical	1
GAZEBO Apartments	HVAC	2
GAZEBO Apartments	Plumbing	2
Oaks of Denton	Appliance Repair	1
Oaks of Denton	Carpentry	1
Oaks of Denton	Electrical	3
Oaks of Denton	HVAC	2
Oaks of Denton	Landscaping	2
Oaks of Denton	Plumbing	1
The Metro Apartments	Appliance Repair	2
The Metro Apartments	Carpentry	2

Result 1

Output

Action Output

#	Time	Action	Message
295	22:50:23	SELECT pm.PropertyName, c.CategoryName, COUNT(mr.RequestID) AS Total_Requests FROM P...	27 row(s) returned

Data analysis 2:

To know the total number of requests which are completed, in progress and open in state based on priority.

SQL Explanation:

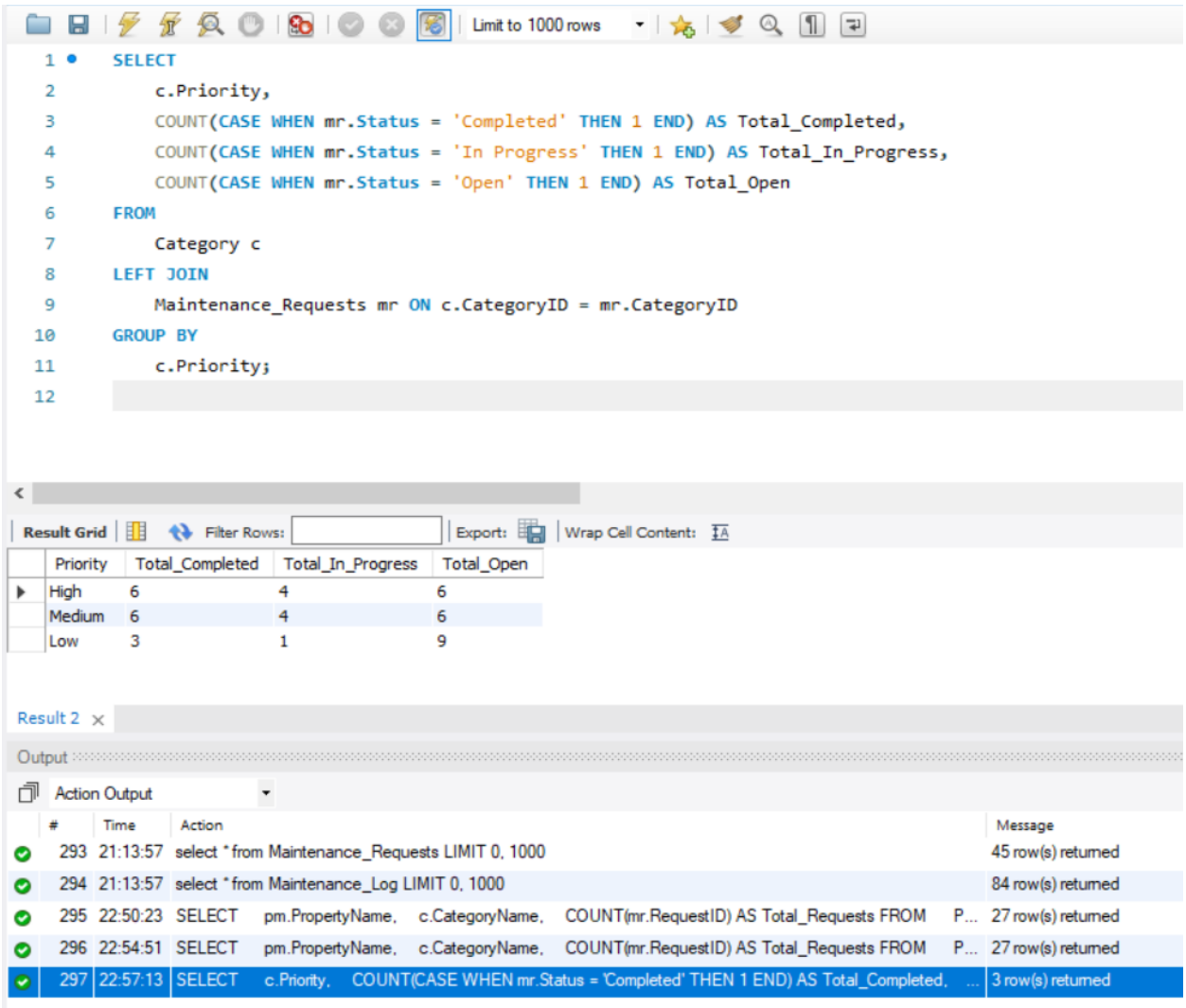
- Selects the PropertyName, CategoryName and Count() (counts the total number of requests for each property based on category) from the join functions.
- Joins the Property_Management as pm, Users as u, Maintenance_Requests as mr and Category as c.

- Joined the table Users with Property_Management table on PropertyID.
- Joined the table Maintenance_Requests with Users table on UsersID.
- Joined the table Maintenance_Requests with Category table on CategoryID.
- Group By groups the results on PropertyName and CategoryName.
- Order By orders the results on PropertyName and CategoryName.

SQL Query:

```
SELECT
   c.Priority,
   COUNT(CASE WHEN mr.Status = 'Completed' THEN 1 END) AS Total_Completed,
   COUNT(CASE WHEN mr.Status = 'In Progress' THEN 1 END) AS Total_In_Progress,
   COUNT(CASE WHEN mr.Status = 'Open' THEN 1 END) AS Total_Open
FROM
   Category c
LEFT JOIN
   Maintenance_Requests mr ON c.CategoryID = mr.CategoryID
GROUP BY
   c.Priority;
```

Screenshot of query and output:



```
SELECT
    c.Priority,
    COUNT(CASE WHEN mr.Status = 'Completed' THEN 1 END) AS Total_Completed,
    COUNT(CASE WHEN mr.Status = 'In Progress' THEN 1 END) AS Total_In_Progress,
    COUNT(CASE WHEN mr.Status = 'Open' THEN 1 END) AS Total_Open
FROM
    Category c
LEFT JOIN
    Maintenance_Requests mr ON c.CategoryID = mr.CategoryID
GROUP BY
    c.Priority;
```

Result Grid

Priority	Total_Completed	Total_In_Progress	Total_Open
High	6	4	6
Medium	6	4	6
Low	3	1	9

Result 2

Output

Action Output

#	Time	Action	Message
293	21:13:57	select * from Maintenance_Requests LIMIT 0, 1000	45 row(s) returned
294	21:13:57	select * from Maintenance_Log LIMIT 0, 1000	84 row(s) returned
295	22:50:23	SELECT pm.PropertyName, c.CategoryName, COUNT(mr.RequestID) AS Total_Requests FROM P...	27 row(s) returned
296	22:54:51	SELECT pm.PropertyName, c.CategoryName, COUNT(mr.RequestID) AS Total_Requests FROM P...	27 row(s) returned
297	22:57:13	SELECT c.Priority, COUNT(CASE WHEN mr.Status = 'Completed' THEN 1 END) AS Total_Completed, ...	3 row(s) returned

Data analysis 3:

To know the total number of requests opened, in progress and which are completed for each maintenance team.

SQL Explanation:

- Selects the count of requests from Maintenance_Requests as Total_Assigned.
- Counts the number of completed requests using case.
- Counts the number of in progress requests using case.
- Counts the number of open requests using case.
- From the Maintenance_Team table which is Joined with the table Maintenance_Requests based on TeamID.
- LEFT JOIN make sure all the records from the left table Maintenance_Team are included, even if no matches are found in right table Maintenance_Requests.
- GROUP BY groups the results on TeamName from Maintenance_Team table.

SQL Query:

```
SELECT
   mt.TeamName,
   COUNT(mr.RequestID) AS Total_Assigned,
   SUM(CASE WHEN mr.Status = 'Completed' THEN 1 ELSE 0 END) AS Total_Complered,
   SUM(CASE WHEN mr.Status = 'In Progress' THEN 1 ELSE 0 END) AS Total_In_Progress,
   SUM(CASE WHEN mr.Status = 'Open' THEN 1 ELSE 0 END) AS Total_Open
FROM
   Maintenance_Teams mt
LEFT JOIN
   Maintenance_Requests mr ON mt.TeamID = mr.TeamID
GROUP BY
   mt.TeamName;
```

Screenshot of query and output:

```sql
SELECT
    mt.TeamName,
    COUNT(mr.RequestID) AS Total_Assigned,
    SUM(CASE WHEN mr.Status = 'Completed' THEN 1 ELSE 0 END) AS Total_Complered,
    SUM(CASE WHEN mr.Status = 'In Progress' THEN 1 ELSE 0 END) AS Total_In_Progress,
    SUM(CASE WHEN mr.Status = 'Open' THEN 1 ELSE 0 END) AS Total_Open
FROM
    Maintenance_Teams mt
LEFT JOIN
    Maintenance_Requests mr ON mt.TeamID = mr.TeamID
GROUP BY
    mt.TeamName;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

TeamName	Total_Assigned	Total_Complered	Total_In_Progress	Total_Open
Plumbing Team	8	3	2	3
Electrical Team	8	3	2	3
HVAC Team	8	3	2	3
Appliance Repair Team	8	3	2	3
Carpentry Team	8	3	1	4
Landscaping Team	5	0	0	5

Result 3

Output

Action Output

#	Time	Action	Message
294	21:13:57	select * from Maintenance_Log LIMIT 0, 1000	84 row(s) returned
295	22:50:23	SELECT pm.PropertyName, c.CategoryName, COUNT(mr.RequestID) AS Total_Requests FROM P...	27 row(s) returned
296	22:54:51	SELECT pm.PropertyName, c.CategoryName, COUNT(mr.RequestID) AS Total_Requests FROM P...	27 row(s) returned
297	22:57:13	SELECT c.Priority, COUNT(CASE WHEN mr.Status = 'Completed' THEN 1 END) AS Total_Completed, ...	3 row(s) returned
298	22:59:43	SELECT mt.TeamName, COUNT(mr.RequestID) AS Total_Assigned, SUM(CASE WHEN mr.Status = '...	6 row(s) returned

Data analysis 4:

To know the total requests which are completed for each maintenance team.

SQL Explanation:

- Selects the count of requests from Maintenance_Requests as Total_Requests_Solved.
- Counts the number of completed requests.
- From the Maintenance_Team table which is Joined with the table Maintenance_Requests based on TeamID.
- LEFT JOIN make sure all the records from the left table Maintenance_Requests are included, even if no matches are found in right table Maintenance_Teams.

- WHERE status of Maintenance_Requests are Completed.
- GROUP BY groups the results on TeamName from Maintenance_Team table.

SQL Query:

```
SELECT
   mt.TeamName,
   COUNT(mr.RequestID) AS Total_Requests_Solved
FROM
   Maintenance_Teams mt
LEFT JOIN
   Maintenance_Requests mr ON mt.TeamID = mr.TeamID
WHERE
   mr.Status = 'Completed'
GROUP BY
   mt.TeamName;
```

Screenshot of query and result:

```sql
SELECT
    mt.TeamName,
    COUNT(mr.RequestID) AS Total_Requests_Solved
FROM
    Maintenance_Teams mt
LEFT JOIN
    Maintenance_Requests mr ON mt.TeamID = mr.TeamID
WHERE
    mr.Status = 'Completed'
GROUP BY
    mt.TeamName;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

TeamName	Total_Requests_Solved
Plumbing Team	3
Electrical Team	3
HVAC Team	3
Appliance Repair Team	3
Carpentry Team	3

Result 4

Output

Action Output

#	Time	Action		Message
297	22:57:13	SELECT	c.Priority, COUNT(CASE WHEN mr.Status = 'Completed' THEN 1 END) AS Total_Completed, ...	3 row(s) returned
298	22:59:43	SELECT	mt.TeamName, COUNT(mr.RequestID) AS Total_Assigned, SUM(CASE WHEN mr.Status = '...	6 row(s) returned
299	23:04:51	SELECT	mt.TeamName, COUNT(mr.RequestID) AS Total_Requests_Solved FROM Maintenance_Te...	5 row(s) returned

Data analysis 5:

To know the properties with most requests raised.

SQL Explanation:

- Selects the property id, property name from Property_Management table and count of total requests as Total_Requests.
- From the Property_Management table which is Joined with the table Users based on PropertyID.
- LEFT JOIN make sure all the records from the left table Users are included, even if no matches are found in right table Maintenance_Requests based on UserID.
- GROUP BY groups the results on PropertyID and PropertyName.
- ORDER BY order the results on Total_Requests in descending order.

SQL Query:

```
SELECT
   p.PropertyID,
   p.PropertyName,
   COUNT(mr.RequestID) AS Total_Requests
FROM
   Property_Management p
LEFT JOIN
   Users u ON p.PropertyID = u.PropertyID
LEFT JOIN
   Maintenance_Requests mr ON u.UserID = mr.UserID
GROUP BY
   p.PropertyID, p.PropertyName
ORDER BY
   Total_Requests DESC;
```

Screenshot of query and output:

```sql
SELECT
    p.PropertyID,
    p.PropertyName,
    COUNT(mr.RequestID) AS Total_Requests
FROM
    Property_Management p
LEFT JOIN
    Users u ON p.PropertyID = u.PropertyID
LEFT JOIN
    Maintenance_Requests mr ON u.UserID = mr.UserID
GROUP BY
    p.PropertyID, p.PropertyName
ORDER BY
    Total_Requests DESC;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

PropertyID	PropertyName	Total_Requests
2	GAZEBO Apartments	11
4	The Metro Apartments	11
3	Oaks of Denton	10
1	The VENUE Apartments	8
5	Epoch on Eagle	5

Result 6

Output

Action Output

#	Time	Action					Message
301	23:48:30	SELECT	p.PropertyID,	p.PropertyName,	COUNT(mr.RequestID) AS Total_Requests FROM	Propert...	5 row(s) returned
302	23:48:51	SELECT	p.PropertyID,	p.PropertyName,	COUNT(mr.RequestID) AS Total_Requests FROM	Propert...	5 row(s) returned

Data analysis 6:

To know the top 10 users with most requests.

SQL Explanation:

- Selects the UserID, Name as UserName from Users table and count of total requests as Total_Requests.
- From the Users table as u which is left joined with the table Maintenance_Requests as mr based on UserID.
- GROUP BY groups the results on UserID and Name of the Users table.
- ORDER BY order the results on Total_Requests in descending order.
- LIMIT the output by 10.

SQL Query:

```
SELECT
   u.UserID,
   u.Name AS UserName,
   COUNT(mr.RequestID) AS TotalRequests
FROM
   Users u
LEFT JOIN
   Maintenance_Requests mr ON u.UserID = mr.UserID
GROUP BY
   u.UserID, u.Name
ORDER BY
   TotalRequests DESC
LIMIT 10;
```

Screenshot of query and output:

```sql
SELECT
    u.UserID,
    u.Name AS UserName,
    COUNT(mr.RequestID) AS TotalRequests
FROM
    Users u
LEFT JOIN
    Maintenance_Requests mr ON u.UserID = mr.UserID
GROUP BY
    u.UserID, u.Name
ORDER BY
    TotalRequests DESC
LIMIT 10;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

UserID	UserName	TotalRequests
1100031	Sophia Hernandez	2
1100073	Sophia Fernandez	2
1100001	Jessica Smith	1
1100022	David Martinez	1
1100003	Emily Brown	1
1100004	Christopher Davis	1
1100006	Matthew Martinez	1
1100009	Lauren Thomas	1
1100010	Ryan Jackson	1
1100011	Ashley White	1

Result 9

Output

Action Output

#	Time	Action	Message
305	23:54:31	SELECT u.UserID, u.Name AS UserName, COUNT(mr.RequestID) AS TotalRequests FROM Users...	10 row(s) returned
306	23:54:35	SELECT u.UserID, u.Name AS UserName, COUNT(mr.RequestID) AS TotalRequests FROM Users...	10 row(s) returned

Appendix:

```
CREATE SCHEMA MaintainEase;
USE MaintainEase;

-- Create Property_Management table
CREATE TABLE Property_Management (
    PropertyID INT AUTO_INCREMENT PRIMARY KEY,
    PropertyName VARCHAR(50) NOT NULL,
    Address VARCHAR(255) NOT NULL, -- Changed from Location
    ManagerName VARCHAR(25) NOT NULL,
    ManagerMail VARCHAR(50) NOT NULL
);

-- Create Users table
CREATE TABLE Users (
    UserID INT AUTO_INCREMENT PRIMARY KEY,
    Name VARCHAR(25) NOT NULL,
    Email VARCHAR(50) NOT NULL UNIQUE, -- Added UNIQUE constraint
    PhoneNo INT,
    LeaseAgreement VARCHAR(255),
    LeaseStartDate DATE,
    LeaseEndDate DATE,
    PropertyID INT,
    FOREIGN KEY (PropertyID) REFERENCES Property_Management(PropertyID)
);

-- Create Category table
CREATE TABLE Category (
    CategoryID INT AUTO_INCREMENT PRIMARY KEY,
```

```
    CategoryName VARCHAR(255) NOT NULL,
    Priority VARCHAR(10) NOT NULL CHECK (Priority IN ('Low', 'Medium', 'High')) -- Added CHECK constraint
);

-- Create Maintenance_Teams table
CREATE TABLE Maintenance_Teams (
    TeamID INT AUTO_INCREMENT PRIMARY KEY,
    TeamName VARCHAR(25) NOT NULL,
    Location VARCHAR(255) NOT NULL
);

-- Create Technicians table
CREATE TABLE Technicians (
    TechnicianID INT AUTO_INCREMENT PRIMARY KEY,
    TeamID INT,
    Name VARCHAR(25) NOT NULL,
    Role VARCHAR(20),
    FOREIGN KEY (TeamID) REFERENCES Maintenance_Teams(TeamID)
);
-- Create Maintenance_Requests table
CREATE TABLE Maintenance_Requests (
    RequestID INT AUTO_INCREMENT PRIMARY KEY,
    UserID INT,
    CategoryID INT,
    TeamID INT,
    RequestRaisedDate DATE NOT NULL,
    RequestCompletedDate DATE,
    Status VARCHAR(20) NOT NULL CHECK (Status IN ('Open', 'In Progress', 'Completed')), -- Added CHECK constraint
    Location VARCHAR(255) NOT NULL,
```

```
    Description TEXT NOT NULL,
    FOREIGN KEY (UserID) REFERENCES Users(UserID),
    FOREIGN KEY (CategoryID) REFERENCES Category(CategoryID),
    FOREIGN KEY (TeamID) REFERENCES Maintenance_Teams(TeamID)
);

-- Create Maintenance_Log table
CREATE TABLE Maintenance_Log (
    LogID INT AUTO_INCREMENT PRIMARY KEY,
    RequestID INT,
    Action VARCHAR(255) NOT NULL,
    Timestamp DATETIME DEFAULT CURRENT_TIMESTAMP, -- Changed to DATETIME
    FOREIGN KEY (RequestID) REFERENCES Maintenance_Requests(RequestID)
);



-- Insert command into property management table
INSERT INTO Property_Management (PropertyID, PropertyName, Address, ManagerName, ManagerMail)
VALUES
(1,'The VENUE Apartments', '1408 Bernard St, Denton, 76201', 'Salena Gomex', 'salena.gomex@gmail.com');
INSERT INTO Property_Management (PropertyName, Address, ManagerName, ManagerMail) VALUES
('GAZEBO Apartments', 'South Carroll Blvd st Denton, 76201', 'Jack Sparrow', 'jack.sparrow@gmail.com'),
('Oaks of Denton', '425 Bernard St, Denton, TX 76201', 'Lucy Fraser', 'lucy.fracer@gmail.com'),
('The Metro Apartments', '627 Bernard St, Denton, TX 76201', 'Greg Marcus', 'greg.marcus@gmail.com'),
('Epoch on Eagle', '903 Avenue C, Denton, TX 76201', 'Shane Watson', 'shane.watson@gmail.com');
select * from Property_Management;
-- Insert data into user table
INSERT INTO Users (UserID, Name, Email, PhoneNo, LeaseAgreement, LeaseStartDate, LeaseEndDate,
PropertyID)
VALUES
```

(1100001,'Jessica Smith', 'jessicasmith@gmail.com', 1234567890, 'LEASE AGREEMENT-VA100', '2023-09-15', '2024-07-01', 1);

ALTER TABLE Users MODIFY PhoneNo BIGINT;

INSERT INTO Users (Name, Email, PhoneNo, LeaseAgreement, LeaseStartDate, LeaseEndDate, PropertyID)

VALUES

('Michael Johnson', 'michaeljohnson@gmail.com', 2345678901, 'LEASE AGREEMENT-VA101', '2023-10-20', '2024-09-10', 1),

('Emily Brown', 'emilybrown@gmail.com', 3456789012, 'LEASE AGREEMENT-VA102', '2023-08-05', '2024-12-15', 1),

('Christopher Davis', 'christopherdavis@gmail.com', 4567890123, 'LEASE AGREEMENT-VA103', '2023-06-25', '2024-11-20', 1),

('Sarah Wilson', 'sarahwilson@gmail.com', 5678901234, 'LEASE AGREEMENT-VA104', '2023-11-10', '2024-08-25', 1),

('Matthew Martinez', 'matthewmartinez@gmail.com', 6789012345, 'LEASE AGREEMENT-VA105', '2023-07-15', '2024-10-05', 1),

('Amanda Anderson', 'amandaanderson@gmail.com', 7890123456, 'LEASE AGREEMENT-VA106', '2023-05-30', '2024-09-30', 1),

('Daniel Taylor', 'danieltaylor@gmail.com', 8901234567, 'LEASE AGREEMENT-VA107', '2023-10-05', '2024-12-10', 1),

('Lauren Thomas', 'laurenthomas@gmail.com', 9012345678, 'LEASE AGREEMENT-VA108', '2023-06-15', '2024-10-20', 1),

('Ryan Jackson', 'ryanjackson@gmail.com', 1236786490, 'LEASE AGREEMENT-VA109', '2023-08-20', '2024-11-15', 1),

('Ashley White', 'ashleywhite@gmail.com', 2356789014, 'LEASE AGREEMENT-VA110', '2023-07-05', '2024-08-15', 1),

('John Harris', 'johnharris@gmail.com', 3478901256, 'LEASE AGREEMENT-VA111', '2023-09-10', '2024-09-05', 1),

('Megan Clark', 'meganclark@gmail.com', 4567012389, 'LEASE AGREEMENT-VA112', '2023-06-30', '2024-12-01', 1),

('Andrew Lewis', 'andrewlewis@gmail.com', 5690123478, 'LEASE AGREEMENT-VA113', '2023-11-05', '2024-08-10', 1),

('Olivia Lee', 'oliviale@gmail.com', 6782345901, 'LEASE AGREEMENT-VA114', '2023-08-10', '2024-10-15', 1),

('James Young', 'jamesyoung@gmail.com', 7123456890, 'LEASE AGREEMENT-VA115', '2023-05-20', '2024-11-30', 1),

('Jessica Turner', 'jessicaturner@gmail.com', 8034567912, 'LEASE AGREEMENT-VA116', '2023-10-15', '2024-09-20', 1),

('William Rodriguez', 'williamrodriguez@gmail.com', 9034578126, 'LEASE AGREEMENT-VA117', '2023-07-25', '2024-08-30', 1),

('Samantha Walker', 'samanthawalker@gmail.com', 1236790458, 'LEASE AGREEMENT-VA118', '2023-09-30', '2024-10-10', 1),

('Brandon Evans', 'brandonevans@gmail.com', 2457901368, 'LEASE AGREEMENT-VA119', '2023-08-05', '2024-12-05', 1),

('Jennifer Garcia', 'jennifergarcia@gmail.com', 1234678905, 'LEASE AGREEMENT-GA100', '2023-09-15', '2024-07-01', 2),

('David Martinez', 'davidmartinez@gmail.com', 2456789013, 'LEASE AGREEMENT-GA101', '2023-10-20', '2024-09-10', 2),

('Mary Lopez', 'marylopez@gmail.com', 3567890124, 'LEASE AGREEMENT-GA102', '2023-08-05', '2024-12-15', 2),

('Jose Gonzalez', 'josegonzalez@gmail.com', 4678901235, 'LEASE AGREEMENT-GA103', '2023-06-25', '2024-11-20', 2),

('Maria Rodriguez', 'mariarodriguez@gmail.com', 5789012346, 'LEASE AGREEMENT-GA104', '2023-11-10', '2024-08-25', 2),

('James Martinez', 'jamesmartinez@gmail.com', 6890123457, 'LEASE AGREEMENT-GA105', '2023-07-15', '2024-10-05', 2),

('Jessica Hernandez', 'jessicahernandez@gmail.com', 7901234568, 'LEASE AGREEMENT-GA106', '2023-05-30', '2024-09-30', 2),

('Daniel Garcia', 'danielgarcia@gmail.com', 8012345679, 'LEASE AGREEMENT-GA107', '2023-10-05', '2024-12-10', 2),

('Emma Lopez', 'emmalopez@gmail.com', 9023456781, 'LEASE AGREEMENT-GA108', '2023-06-15', '2024-10-20', 2),

('Alexander Martinez', 'alexandermartinez@gmail.com', 1235678904, 'LEASE AGREEMENT-GA109', '2023-08-20', '2024-11-15', 2),

('Sophia Hernandez', 'sophiahernandez@gmail.com', 2456780139, 'LEASE AGREEMENT-GA110', '2023-07-05', '2024-08-15', 2),

('Jacob Gonzalez', 'jacobgonzalez@gmail.com', 3468012597, 'LEASE AGREEMENT-GA111', '2023-09-10', '2024-09-05', 2),

('Isabella Rodriguez', 'isabellarodriguez@gmail.com', 4567823190, 'LEASE AGREEMENT-GA112', '2023-06-30', '2024-12-01', 2),

('William Garcia', 'williamgarcia@gmail.com', 5791234806, 'LEASE AGREEMENT-GA113', '2023-11-05', '2024-08-10', 2),

('Sophia Martinez', 'sophiamartinez@gmail.com', 6781245309, 'LEASE AGREEMENT-GA114', '2023-08-10', '2024-10-15', 2),

('Daniel Fernandez', 'danielfernandez@gmail.com', 7893456102, 'LEASE AGREEMENT-GA115', '2023-05-20', '2024-11-30', 2),

('Olivia Lopez', 'olivialopez@gmail.com', 8901234576, 'LEASE AGREEMENT-GA116', '2023-10-15', '2024-09-20', 2),

('Christopher Hernandez', 'christopherhernandez@gmail.com', 9012346678, 'LEASE AGREEMENT-GA117', '2023-07-25', '2024-08-30', 2),

('Emily Gonzalez', 'emilygonzalez@gmail.com', 1234567890, 'LEASE AGREEMENT-GA118', '2023-09-30', '2024-10-10', 2),

('Michael Martinez', 'michaelmartinez@gmail.com', 2356789140, 'LEASE AGREEMENT-GA119', '2023-08-05', '2024-12-05', 2),

('Emma Walker', 'emmawalker@gmail.com', 2345789016, 'LEASE AGREEMENT-OD101', '2023-10-20', '2024-09-10', 3),

('James Harris', 'jamesharris@gmail.com', 3467890125, 'LEASE AGREEMENT-OD102', '2023-08-05', '2024-12-15', 3),

('Olivia Robinson', 'oliviarobinson@gmail.com', 4578901236, 'LEASE AGREEMENT-OD103', '2023-06-25', '2024-11-20', 3),

('Lucas Young', 'lucasyoung@gmail.com', 5689012347, 'LEASE AGREEMENT-OD104', '2023-11-10', '2024-08-25', 3),

('Ava Martinez', 'avamartinez@gmail.com', 6790123458, 'LEASE AGREEMENT-OD105', '2023-07-15', '2024-10-05', 3),

('Daniel Hernandez', 'danielhernandez@gmail.com', 7801234569, 'LEASE AGREEMENT-OD106', '2023-05-30', '2024-09-30', 3),

('Sophia Diaz', 'sophiadiaz@gmail.com', 8901245673, 'LEASE AGREEMENT-OD107', '2023-10-05', '2024-12-10', 3),

('Jacob Rodriguez', 'jacobrodriguez@gmail.com', 9012456783, 'LEASE AGREEMENT-OD108', '2023-06-15', '2024-10-20', 3),

('Isabella Martinez', 'isabellamartinez@gmail.com', 1245678903, 'LEASE AGREEMENT-OD109', '2023-08-20', '2024-11-15', 3),

('Alexander Lopez', 'alexanderlopez@gmail.com', 2345689017, 'LEASE AGREEMENT-OD110', '2023-07-05', '2024-08-15', 3),

('Mia Gonzalez', 'miagonzalez@gmail.com', 3468012579, 'LEASE AGREEMENT-OD111', '2023-09-10', '2024-09-05', 3),

('Ethan Perez', 'ethanperez@gmail.com', 4567823901, 'LEASE AGREEMENT-OD112', '2023-06-30', '2024-12-01', 3),

('Sophia Smith', 'sophiasmith@gmail.com', 5791234680, 'LEASE AGREEMENT-OD113', '2023-11-05', '2024-08-10', 3),

('Jackson Anson', 'jacksonanson@gmail.com', 6781245390, 'LEASE AGREEMENT-OD114', '2023-08-10', '2024-10-15', 3),

('Charlotte Wilson', 'charlottewilson@gmail.com', 7893456201, 'LEASE AGREEMENT-OD115', '2023-05-20', '2024-11-30', 3),

('Logan Taylor', 'logantaylor@gmail.com', 8901246754, 'LEASE AGREEMENT-OD116', '2023-10-15', '2024-09-20', 3),

('Amelia White', 'ameliawhite@gmail.com', 9012357864, 'LEASE AGREEMENT-OD117', '2023-07-25', '2024-08-30', 3),

('Mason Jackson', 'masonjackson@gmail.com', 4567890321, 'LEASE AGREEMENT-OD118', '2023-09-30', '2024-10-10', 3),

('Ell Haris', 'ellharis@gmail.com', 2345678109, 'LEASE AGREEMENT-OD119', '2023-08-05', '2024-12-05', 3),

('Sophia Brown', 'sophiabrown@gmail.com', 2345679018, 'LEASE AGREEMENT-MA101', '2023-10-20', '2024-09-10', 4),

('William Davis', 'williamdavis@gmail.com', 3457890126, 'LEASE AGREEMENT-MA102', '2023-08-05', '2024-12-15', 4),

('Isabella Miller', 'isabellamiller@gmail.com', 4568901237, 'LEASE AGREEMENT-MA103', '2023-06-25', '2024-11-20', 4),

('Mason Wilson', 'masonwilson@gmail.com', 5679012348, 'LEASE AGREEMENT-MA104', '2023-11-10', '2024-08-25', 4),

('Emma Moore', 'emmamoore@gmail.com', 6780123459, 'LEASE AGREEMENT-MA105', '2023-07-15', '2024-10-05', 4),

('Olivia Taylor', 'oliviataylor@gmail.com', 7891234560, 'LEASE AGREEMENT-MA106', '2023-05-30', '2024-09-30', 4),

('Jackson Anderson', 'jacksonanderson@gmail.com', 8902345671, 'LEASE AGREEMENT-MA107', '2023-10-05', '2024-12-10', 4),

('Ava Thomas', 'avathomas@gmail.com', 9012346785, 'LEASE AGREEMENT-MA108', '2023-06-15', '2024-10-20', 4),

('James White', 'jameswhite@gmail.com', 1345678902, 'LEASE AGREEMENT-MA109', '2023-08-20', '2024-11-15', 4),

('Ella Harris', 'ellaharris@gmail.com', 2345678019, 'LEASE AGREEMENT-MA110', '2023-07-05', '2024-08-15', 4),

('Logan Clark', 'loganclark@gmail.com', 3589012467, 'LEASE AGREEMENT-MA111', '2023-09-10', '2024-09-05', 4),

('Amelia Lee', 'ameliale@gmail.com', 4678023591, 'LEASE AGREEMENT-MA112', '2023-06-30', '2024-12-01', 4),

('Michael Young', 'michaelyoung@gmail.com', 5689234601, 'LEASE AGREEMENT-MA113', '2023-11-05', '2024-08-10', 4),

('Sophia Fernandez', 'sophiafernandez@gmail.com', 6902345187, 'LEASE AGREEMENT-MA114', '2023-08-10', '2024-10-15', 4),

('Daniel Rodriguez', 'danielrodriguez@gmail.com', 7901346825, 'LEASE AGREEMENT-MA115', '2023-05-20', '2024-11-30', 4),

('Aiden Martinez', 'aidenmartinez@gmail.com', 8934567120, 'LEASE AGREEMENT-MA116', '2023-10-15', '2024-09-20', 4),

('Abigail King', 'abigailking@gmail.com', 9012578634, 'LEASE AGREEMENT-MA117', '2023-07-25', '2024-08-30', 4),

('Madison Perez', 'madisonperez@gmail.com', 1234780569, 'LEASE AGREEMENT-MA118', '2023-09-30', '2024-10-10', 4),

('Carter Rivera', 'carterrivera@gmail.com', 2456891037, 'LEASE AGREEMENT-MA119', '2023-08-05', '2024-12-05', 4),

('Charlotte Walker', 'charlottewalker@gmail.com', 2345678910, 'LEASE AGREEMENT-EE101', '2023-10-20', '2024-09-10', 5),

('William Harris', 'williamharris@gmail.com', 3456890127, 'LEASE AGREEMENT-EE102', '2023-08-05', '2024-12-15', 5),

('Sophia Robinson', 'sophiarobinson@gmail.com', 4567901238, 'LEASE AGREEMENT-EE103', '2023-06-25', '2024-11-20', 5),

('Logan Young', 'loganyoung@gmail.com', 5678012349, 'LEASE AGREEMENT-EE104', '2023-11-10', '2024-08-25', 5),

('Emma Martinez', 'emmamartinez@gmail.com', 6789123450, 'LEASE AGREEMENT-EE105', '2023-07-15', '2024-10-05', 5),

('Jacob Taylor', 'jacobtaylor@gmail.com', 7890234561, 'LEASE AGREEMENT-EE106', '2023-05-30', '2024-09-30', 5),

('Ava Diaz', 'avadiaz@gmail.com', 8901345672, 'LEASE AGREEMENT-EE107', '2023-10-05', '2024-12-10', 5),

('Michael Rodriguez', 'michaelrodriguez@gmail.com', 9012345687, 'LEASE AGREEMENT-EE108', '2023-06-15', '2024-10-20', 5),

('Isabell Marnez', 'isabellmarnez@gmail.com', 1234568907, 'LEASE AGREEMENT-EE109', '2023-08-20', '2024-11-15', 5),

('Alexander Lepoz', 'alexanderlepoz@gmail.com', 2347890156, 'LEASE AGREEMENT-EE110', '2023-07-05', '2024-08-15', 5),

('Sophia Gonzalez', 'sophiagonzalez@gmail.com', 3467901285, 'LEASE AGREEMENT-EE111', '2023-09-10', '2024-09-05', 5),

('James Perez', 'jamesperez@gmail.com', 4568902371, 'LEASE AGREEMENT-EE112', '2023-06-30', '2024-12-01', 5),

('Emma Smith', 'emmasmith@gmail.com', 5601234111, 'LEASE AGREEMENT-EE113', '2023-11-05', '2024-08-10', 5),

('Jacob Hernandez', 'jacobhernandez@gmail.com', 6789112345, 'LEASE AGREEMENT-EE114', '2023-08-10', '2024-10-15', 5),

('Olivia Rodriguez', 'oliviarodriguez@gmail.com', 7890223456, 'LEASE AGREEMENT-EE115', '2023-05-20', '2024-11-30', 5),

('Ethan Martinez', 'ethanmartinez@gmail.com', 8901334567, 'LEASE AGREEMENT-EE116', '2023-10-15', '2024-09-20', 5),

('Ava White', 'avawhite@gmail.com', 9013345678, 'LEASE AGREEMENT-EE117', '2023-07-25', '2024-08-30', 5),

('William Clark', 'williamclark@gmail.com', 1234667890, 'LEASE AGREEMENT-EE118', '2023-09-30', '2024-10-10', 5),

('Madison Lee', 'madisonlee@gmail.com', 2355678901, 'LEASE AGREEMENT-EE119', '2023-08-05', '2024-12-05', 5);

select * from users;

-- Insert commands for the Category table

INSERT INTO Category (CategoryID, CategoryName, Priority) VALUES (1401, 'Plumbing', 'High');

INSERT INTO Category (CategoryName, Priority)

VALUES

('Electrical', 'High'),

('HVAC', 'Medium'),

('Appliance Repair', 'Medium'),

('Carpentry', 'Low'),

('Landscaping', 'Low');

Select * from Category;

-- Insert commands for the Maintenance_Teams table

INSERT INTO Maintenance_Teams (TeamID, TeamName, Location) VALUES (1501, 'Plumbing Team', '123 Oak Street, Denton, TX');

INSERT INTO Maintenance_Teams (TeamName, Location)

VALUES

('Electrical Team', '456 Maple Avenue, Denton, TX'),

('HVAC Team', '789 Pine Road, Denton, TX'),

('Appliance Repair Team', '987 Cedar Street, Denton, TX'),

('Carpentry Team', '101 Elm Boulevard, Denton, TX'),

```
('Landscaping Team', '321 Cedar Lane, Denton, TX');
select * from Maintenance_Teams;

-- Insert commands for the Technicians table
INSERT INTO Technicians (TechnicianID, TeamID, Name, Role) VALUES
(1601,1501, 'John Smith', 'Plumber');
INSERT INTO Technicians (TeamID, Name, Role) VALUES
(1501, 'Emily Johnson', 'Plumber'),
(1501, 'Michael Williams', 'Plumber'),
(1501, 'Jessica Brown', 'Plumber'),
(1501, 'Christopher Davis', 'Plumber'),
(1502, 'David Jones', 'Electrician'),
(1502, 'Sarah Martinez', 'Electrician'),
(1502, 'Robert Miller', 'Electrician'),
(1502, 'Linda Wilson', 'Electrician'),
(1502, 'Daniel Taylor', 'Electrician'),
(1503, 'William Anderson', 'HVAC Technician'),
(1503, 'Jennifer Garcia', 'HVAC Technician'),
(1503, 'Thomas Lee', 'HVAC Technician'),
(1503, 'Margaret Rodriguez', 'HVAC Technician'),
(1503, 'Richard Hernandez', 'HVAC Technician'),
(1504, 'Jennifer Turner', 'Appliance Technician'),
(1504, 'Brian Scott', 'Appliance Technician'),
(1504, 'Lisa King', 'Appliance Technician'),
(1504, 'Anthony Perez', 'Appliance Technician'),
(1504, 'Karen Green', 'Appliance Technician'),
(1505, 'Mary Martinez', 'Carpenter'),
(1505, 'James Gonzalez', 'Carpenter'),
(1505, 'Patricia Wright', 'Carpenter'),
(1505, 'Charles Lopez', 'Carpenter'),
```

```
(1505, 'Angela Hill', 'Carpenter'),
(1506, 'Mark Young', 'Landscaper'),
(1506, 'Anna Moore', 'Landscaper'),
(1506, 'Joseph Clark', 'Landscaper'),
(1506, 'Elizabeth Lewis', 'Landscaper'),
(1506, 'Kevin Hall', 'Landscaper');

select * from Technicians;

-- Create a trigger to log entry request actions on Maintenance_Requests table

DELIMITER //

CREATE TRIGGER maintenance_request_insert_trigger
AFTER INSERT ON Maintenance_Requests
FOR EACH ROW
BEGIN
    -- Get the status of the updated request
    DECLARE new_status VARCHAR(20);
    SELECT Status INTO new_status FROM Maintenance_Requests WHERE RequestID = NEW.RequestID;

    -- Insert a record into Maintenance_Log for each action
    INSERT INTO Maintenance_Log (RequestID, Action, Timestamp)
    VALUES (NEW.RequestID, CONCAT('Status changed to: ', new_status), NOW());
END//

DELIMITER ;

-- Insert commands for the Maintenance_Requests table
INSERT INTO Maintenance_Requests (RequestID, UserID, CategoryID, TeamID, Location, Description,
RequestRaisedDate, Status) VALUES
```

(1201, 1100001, 1401, 1501, '1408 Bernard St, Denton, 76201', 'Leaky faucet in the kitchen', '2024-04-27', 'Open'),

(1202, 1100003, 1402, 1502, '1408 Bernard St, Denton, 76201', 'Light fixture not working in the living room', '2024-04-27', 'Open'),

(1203, 1100004, 1403, 1503, '1408 Bernard St, Denton, 76201', 'AC unit not cooling properly', '2024-04-27', 'Open'),

(1204, 1100006, 1404, 1504, '1408 Bernard St, Denton, 76201', 'Door hinge loose in the bedroom', '2024-04-27', 'Open'),

(1205, 1100009, 1405, 1505, '1408 Bernard St, Denton, 76201', 'Grass needs cutting in the backyard', '2024-04-27', 'Open'),

(1206, 1100010, 1401, 1501, '1408 Bernard St, Denton, 76201', 'Sink clogged in the bathroom', '2024-04-27', 'Open'),

(1207, 1100018, 1402, 1502, '1408 Bernard St, Denton, 76201', 'Outlet not working in the kitchen', '2024-04-27', 'Open'),

(1208, 1100022, 1403, 1503, 'South Carroll Blvd st Denton, 76201', 'AC unit making strange noise', '2024-04-27', 'Open'),

(1209, 1100027, 1404, 1504, 'South Carroll Blvd st Denton, 76201', 'Drawer stuck in the living room', '2024-04-27', 'Open'),

(1210, 1100023, 1405, 1505, 'South Carroll Blvd st Denton, 76201', 'Shrubbery needs trimming in the front yard', '2024-04-27', 'Open'),

(1211, 1100025, 1401, 1501, 'South Carroll Blvd st Denton, 76201', 'Toilet running in the bathroom', '2024-04-27', 'Open'),

(1212, 1100029, 1402, 1502, 'South Carroll Blvd st Denton, 76201', 'Ceiling fan not turning on in the bedroom', '2024-04-27', 'Open'),

(1213, 1100031, 1403, 1503, 'South Carroll Blvd st Denton, 76201', 'Heater not heating properly in the living room', '2024-04-27', 'Open'),

(1214, 1100033, 1404, 1504, 'South Carroll Blvd st Denton, 76201', 'Closet door off its track in the hallway', '2024-04-27', 'Open'),

(1215, 1100038, 1405, 1505, 'South Carroll Blvd st Denton, 76201', 'Garden bed needs weeding in the backyard', '2024-04-27', 'Open'),

(1216, 1100037, 1401, 1501, 'South Carroll Blvd st Denton, 76201', 'Shower head leaking in the bathroom', '2024-04-27', 'Open'),

(1217, 1100041, 1402, 1502, '425 Bernard St, Denton, TX 76201', 'Light flickering in the dining room', '2024-04-27', 'Open'),

(1218, 1100044, 1403, 1503, '425 Bernard St, Denton, TX 76201', 'AC unit not turning on in the office', '2024-04-27', 'Open'),

(1219, 1100047, 1404, 1504, '425 Bernard St, Denton, TX 76201', 'Cabinet door broken in the kitchen', '2024-04-27', 'Open'),

(1220, 1100052, 1405, 1505, '425 Bernard St, Denton, TX 76201', 'Mulch needs replenishing in the front yard', '2024-04-27', 'Open'),

(1221, 1100053, 1401, 1501, '425 Bernard St, Denton, TX 76201', 'Garbage disposal jammed in the kitchen', '2024-04-27', 'Open'),

(1222, 1100056, 1402, 1502, '425 Bernard St, Denton, TX 76201', 'Bathroom exhaust fan not working in the bathroom', '2024-04-27', 'Open'),

(1223, 1100055, 1403, 1503, '425 Bernard St, Denton, TX 76201', 'Thermostat not responding in the living room', '2024-04-27', 'Open'),

(1224, 1100066, 1404, 1504, '627 Bernard St, Denton, TX 76201', 'Drawer handle loose in the bedroom', '2024-04-27', 'Open'),

(1225, 1100067, 1405, 1505, '627 Bernard St, Denton, TX 76201', 'Lawn needs mowing in the backyard', '2024-04-27', 'Open'),

(1226, 1100069, 1401, 1501, '627 Bernard St, Denton, TX 76201', 'Leak under the sink in the kitchen', '2024-04-27', 'Open'),

(1227, 1100070, 1402, 1502, '627 Bernard St, Denton, TX 76201', 'Outlet sparking in the living room', '2024-04-27', 'Open'),

(1228, 1100071, 1403, 1503, '627 Bernard St, Denton, TX 76201', 'AC unit leaking water in the bedroom', '2024-04-27', 'Open'),

(1229, 1100075, 1404, 1504, '627 Bernard St, Denton, TX 76201', 'Closet door stuck in the hallway', '2024-04-27', 'Open'),

(1230, 1100072, 1405, 1505, '627 Bernard St, Denton, TX 76201', 'Tree limb needs trimming in the backyard', '2024-04-27', 'Open'),

(1231, 1100073, 1401, 1501, '627 Bernard St, Denton, TX 76201', 'Toilet not flushing in the bathroom', '2024-04-27', 'Open'),

(1232, 1100081, 1402, 1502, '903 Avenue C, Denton, TX 76201', 'Ceiling light not turning off in the bedroom', '2024-04-27', 'Open'),

(1233, 1100061, 1403, 1503, '627 Bernard St, Denton, TX 76201', 'Heater blowing cold air in the living room', '2024-04-27', 'Open'),

(1234, 1100031, 1404, 1504, 'South Carroll Blvd st Denton, 76201', 'Drawer wont close in the kitchen', '2024-04-27', 'Open'),

(1235, 1100021, 1405, 1505, 'South Carroll Blvd st Denton, 76201', 'Bushes need trimming in the front yard', '2024-04-27', 'Open'),

(1236, 1100011, 1401, 1501, '1408 Bernard St, Denton, 76201', 'Sink faucet dripping in the bathroom', '2024-04-27', 'Open'),

(1237, 1100051, 1402, 1502, '425 Bernard St, Denton, TX 76201', 'Dimmer switch not working in the dining room', '2024-04-27', 'Open'),

```
(1238, 1100088, 1403, 1503, '903 Avenue C, Denton, TX 76201', 'AC unit blowing warm air in the office', '2024-04-27', 'Open'),

(1239, 1100083, 1404, 1504, '903 Avenue C, Denton, TX 76201', 'Shelf bracket broken in the kitchen', '2024-04-27', 'Open'),

(1240, 1100084, 1405, 1505, '903 Avenue C, Denton, TX 76201', 'Weeds need pulling in the backyard', '2024-04-27', 'Open'),

(1241, 1100086, 1406, 1506, '903 Avenue C, Denton, TX 76201', 'Refrigerator not cooling properly', '2024-04-27', 'Open'),

(1242, 1100054, 1406, 1506, '425 Bernard St, Denton, TX 76201', 'Dishwasher not draining', '2024-04-27', 'Open'),

(1243, 1100050, 1406, 1506, '425 Bernard St, Denton, TX 76201', 'Oven not heating', '2024-04-27', 'Open'),

(1244, 1100068, 1406, 1506, '627 Bernard St, Denton, TX 76201', 'Washing machine not spinning', '2024-04-27', 'Open'),

(1245, 1100073, 1406, 1506, '627 Bernard St, Denton, TX 76201', 'Dryer not drying clothes', '2024-04-27', 'Open');


select * from Maintenance_Requests;


-- Create a trigger to log update request info actions on Maintenance_Requests table


DELIMITER //


CREATE TRIGGER maintenance_request_update_trigger

AFTER UPDATE ON Maintenance_Requests

FOR EACH ROW

BEGIN

    -- Get the status of the updated request

    DECLARE new_status VARCHAR(20);

    SELECT Status INTO new_status FROM Maintenance_Requests WHERE RequestID = NEW.RequestID;


    -- Insert a record into Maintenance_Log for each action

    INSERT INTO Maintenance_Log (RequestID, Action, Timestamp)

    VALUES (NEW.RequestID, CONCAT('Status changed to: ', new_status), NOW());

END//
```

```
DELIMITER ;



UPDATE Maintenance_Requests SET Status = 'In Progress' WHERE RequestID = 1201;
UPDATE Maintenance_Requests SET Status = 'In Progress' WHERE RequestID = 1202;
UPDATE Maintenance_Requests SET Status = 'In Progress' WHERE RequestID = 1203;
UPDATE Maintenance_Requests SET Status = 'In Progress' WHERE RequestID = 1204;
UPDATE Maintenance_Requests SET Status = 'In Progress' WHERE RequestID = 1205;
UPDATE Maintenance_Requests SET Status = 'In Progress' WHERE RequestID = 1206;
UPDATE Maintenance_Requests SET Status = 'In Progress' WHERE RequestID = 1207;
UPDATE Maintenance_Requests SET Status = 'In Progress' WHERE RequestID = 1208;
UPDATE Maintenance_Requests SET Status = 'In Progress' WHERE RequestID = 1209;
UPDATE Maintenance_Requests SET Status = 'In Progress' WHERE RequestID = 1210;
UPDATE Maintenance_Requests SET Status = 'In Progress' WHERE RequestID = 1211;
UPDATE Maintenance_Requests SET Status = 'In Progress' WHERE RequestID = 1212;
UPDATE Maintenance_Requests SET Status = 'In Progress' WHERE RequestID = 1213;
UPDATE Maintenance_Requests SET Status = 'In Progress' WHERE RequestID = 1214;
UPDATE Maintenance_Requests SET Status = 'In Progress' WHERE RequestID = 1215;
UPDATE Maintenance_Requests SET Status = 'In Progress' WHERE RequestID = 1216;
UPDATE Maintenance_Requests SET Status = 'In Progress' WHERE RequestID = 1217;
UPDATE Maintenance_Requests SET Status = 'In Progress' WHERE RequestID = 1218;
UPDATE Maintenance_Requests SET Status = 'In Progress' WHERE RequestID = 1219;
UPDATE Maintenance_Requests SET Status = 'In Progress' WHERE RequestID = 1220;
UPDATE Maintenance_Requests SET Status = 'In Progress' WHERE RequestID = 1221;
UPDATE Maintenance_Requests SET Status = 'In Progress' WHERE RequestID = 1222;
UPDATE Maintenance_Requests SET Status = 'In Progress' WHERE RequestID = 1223;
UPDATE Maintenance_Requests SET Status = 'In Progress' WHERE RequestID = 1224;

Select * from Maintenance_Requests;
```

```
Select * from Maintenance_Log;

UPDATE Maintenance_Requests SET Status = 'Completed' WHERE RequestID = 1201;
UPDATE Maintenance_Requests SET Status = 'Completed' WHERE RequestID = 1202;
UPDATE Maintenance_Requests SET Status = 'Completed' WHERE RequestID = 1203;
UPDATE Maintenance_Requests SET Status = 'Completed' WHERE RequestID = 1204;
UPDATE Maintenance_Requests SET Status = 'Completed' WHERE RequestID = 1205;
UPDATE Maintenance_Requests SET Status = 'Completed' WHERE RequestID = 1206;
UPDATE Maintenance_Requests SET Status = 'Completed' WHERE RequestID = 1207;
UPDATE Maintenance_Requests SET Status = 'Completed' WHERE RequestID = 1208;
UPDATE Maintenance_Requests SET Status = 'Completed' WHERE RequestID = 1209;
UPDATE Maintenance_Requests SET Status = 'Completed' WHERE RequestID = 1210;
UPDATE Maintenance_Requests SET Status = 'Completed' WHERE RequestID = 1211;
UPDATE Maintenance_Requests SET Status = 'Completed' WHERE RequestID = 1212;
UPDATE Maintenance_Requests SET Status = 'Completed' WHERE RequestID = 1213;
UPDATE Maintenance_Requests SET Status = 'Completed' WHERE RequestID = 1214;
UPDATE Maintenance_Requests SET Status = 'Completed' WHERE RequestID = 1215;


Select * from Maintenance_Requests;

Select * from Maintenance_Log;
```