IST 659: FINAL PROJECT REPORT:

SYRACUSE APARTMENT RENTING SYSTEM

Abstract: The project is based on centrally managing the different functionalities of housing agency including the lease details and the work order request tracking. It also has reporting capabilities and role-based access enabled with proper authentication implemented at all levels.

Niti Saluja

SUID: 951948321

Table of Contents

Project Summary

With the current paradigm shift in the technological realm, there is an urgent need to embrace the change in all sectors including the Housing sector. Over the years, property owners and rental managers have faced multiple challenges with managing the rental details, complaints and their respective lease agreements. This is primarily due to enormously large amount of data and lack of secured computerized systems available for the real estate agencies. The database system aims at providing an ideal solution to ameliorate the rental managers' pain of managing multiple agreements thereby enabling them to manage all their work efficiently and effectively.

In a city like Syracuse where the primary source of employment comes from the Education sector, it is of utmost importance to provide timely repair services and ensure regular maintenance of utilities for the students. As per the existing process, the housing agencies tend to ignore student's complaints or repair requests at times due to the heavy workload. Also, it is not feasible for the landlord to address the increasing number of requests from the tenants, hence the system will help in tracking them all centrally so that they can be assigned to the employees of the organization thereby simplifying the process. The database system would benefit the property owners of large agencies like Concord housing, University Hill etc. owning multiple apartments, as it is a laborious task for them to keep a track of the leases of all the rentals. It will integrate all the components of apartment renting under a single umbrella thereby streamlining the process for both students and property owners.

The general business function of the database system includes maintaining the lease details of all the tenants centrally as well as managing the work order requests so that they can be worked upon on priority.

Designed Solution:

- The user will login via a web Graphical User Interface (GUI)
- All the users can access the application content based upon the assigned role based access.
- The property owner can retrieve, edit or create new lease, tenant, apartment and work order details. They also have access to the different aggregated analytical reports and graph for high level analysis of data
- The tenants can view or update their profile details and create new work orders for their respective apartments.
- These work orders are auto assigned to the employees based on their job type using a trigger.
- The Employees can view or edit their profile and update the status of the assigned work orders on completion.

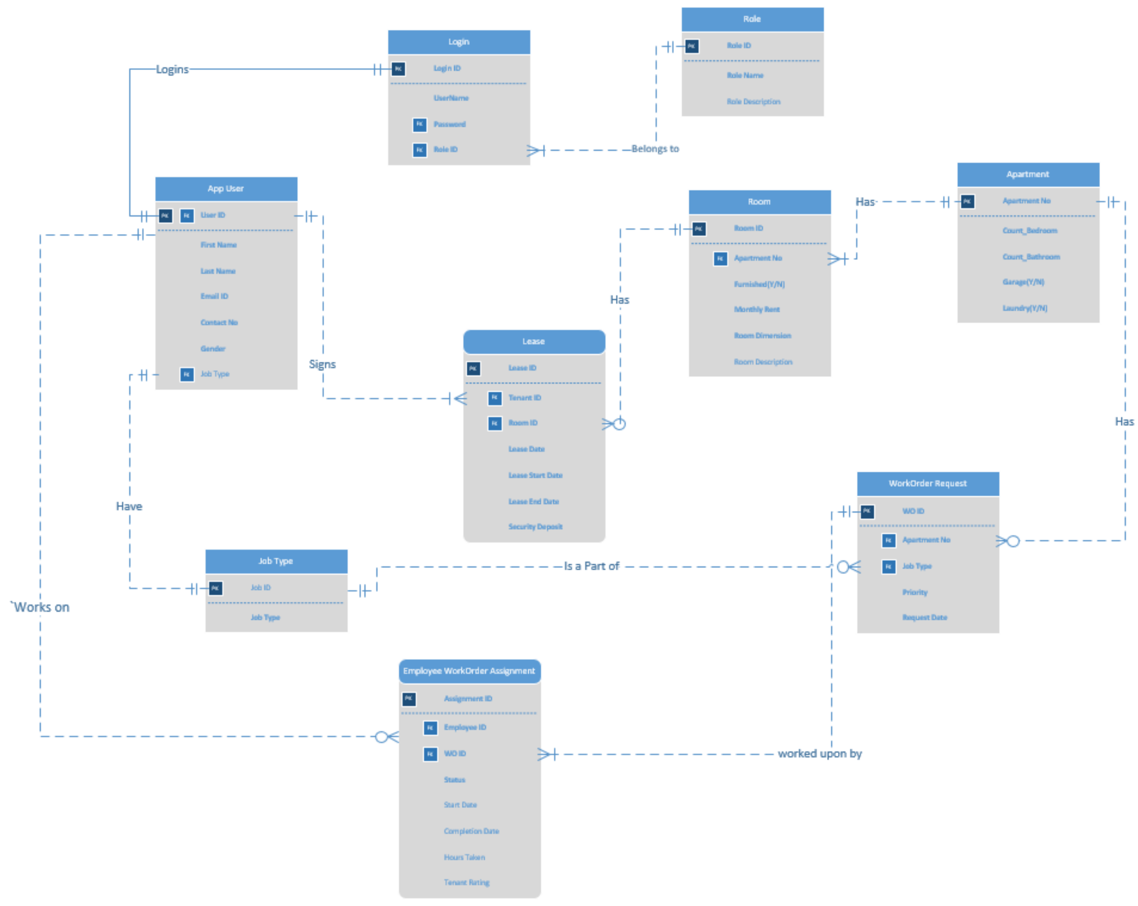
Entity & Attributes:

Data Objects: **Rental Management System**	This Database contains all the tables, their attributes and the relationship that build the rental management system
<u>Proj_Role</u>	Stores the details of the roles present in the application
• Role ID • Role Name • Role Description	Role ID: ID of the role Role Name: Either TENANT, EMPLOYEE or LANDLORD Primary Key- Each role has unique **Role ID,** so it is the primary key
<u>Proj_AppUser</u>	Stores the details of the user
• UserID • FirstName • LastName • EmailID • ContactNo • Gender • Job Type	Primary Key: User ID will uniquely identify each user of the system. The 3 categories of users would be i. Tenants ii. Landlord iii. Employees Foreign Key: Job Type referenced from Proj_JobType table indicating the specialized Job Types of the Employees.
<u>Proj_JobType</u>	Stores the details of the different Job Types
• Job ID • Job Type	Primary Key: Job ID will uniquely identify each of the Job Types
<u>Proj_Lease</u>	Stores the lease details of the tenant
• Lease ID • Room_ID • Tenant ID • Lease Date • Lease Start Date • Lease End Date • Security Deposit	Primary Key: Lease ID is unique for each generated lease of the tenant hence primary key Foreign Key: i) Room ID referenced from Rooms table indicating the room for which the lease is confirmed. ii) Tenant ID referenced from Users table indicating the tenant who signed the lease Lease Date is the date when the lease is confirmed.
<u>Proj_Room</u>	Stores the room details of an apartment
• Room ID • Apartment No • Furnished(Y/N) • Monthly Rent • Room Dimensions • Room Description	Primary Key: Room ID is unique for each room hence the primary key Foreign Key: Apartment No referenced from Apartment table is a required foreign key indicating the apartment to which the room belongs
<u>Proj_Apartment</u>	Stores the apartment details

• Apartment No • Count_Bedroom • Count_Bathroom • Garage(Y/N) • Laundry(Y/N)	Primary Key: Apartment No is unique for the apartments and will unique identify each of them hence the primary key Count_Bedroom: The number of bedrooms in the apartment Count_Bathroom: The number of bathrooms in the apartment
Proj_WorkOrder_Request	Stores the work order requests logged by the tenants
• WO ID • Apartment No • Job Type • Priority • Request Date	Primary Key –WO ID is the unique ID identifying each of the work order requests Foreign Key: i)Apartment No referenced from Apartment table is a required foreign key indicating the apartment no for which the request is logged. ii) Job Type referenced from the Proj_JobType table indicating the Job Type of the request work order. Request Date is the date when the request was logged
Proj_Employee_Assignment	Allows to track the employees working on the work requests
• Assignment ID • Employee ID • WO ID • Status • Completion Date • Start Date • Hours Taken • Tenant Rating	This table acts as an associative table for Users(Employee) and Work Order Request table having a many-to-many relationship Primary Key: Assignment ID is an identifier uniquely identifying each work order assignment combination, hence primary key Foreign Key: i)Employee ID referenced from the Users table indicating the employee working on the request ii)WO ID referenced from the Work Order Request table indicating the Work Order which is been worked upon by the employee

Relational Data Model:

Visio ER Diagram:

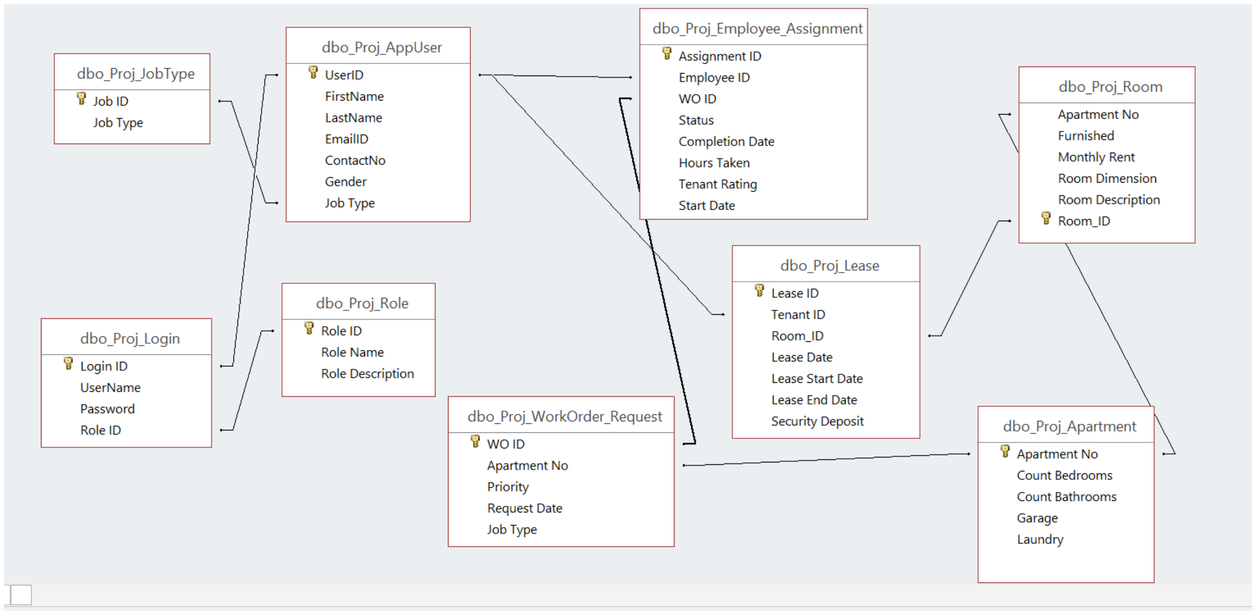


Business Rules:

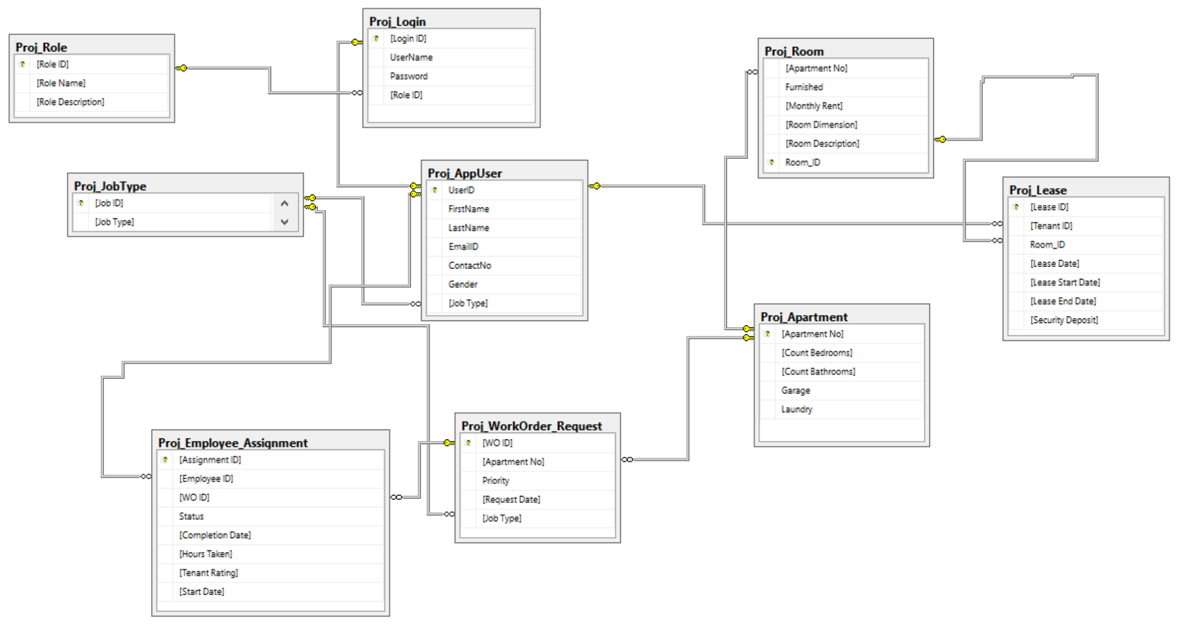
- There are three different user roles – Landlord, Tenant and Employee, they will have separate flow in the application and are treated as separate entities.
- The user details are stored in a single entity and he has role-based access to the system based upon the role attribute stored in the Login entity, which is in turn joined to the User entity.
- Landlord has administrative rights on the system and can manage the lease details of the tenants and track down the logged work order requests along with the assignment details and progress
- Tenants are the users of the systems (mostly students) who have leased a room of the apartment. They can modify or view their details and log work order requests.
- Employee are the workers of the housing agency who work upon the logged work order requests.
- Each App User has one and only one login id to the login page.

- Each Login ID has exactly one assigned role; A role must have at least one Login ID . Here I have assumed that a role has minimum one user.
- Each user (tenant) owns one or more leases; Each lease is owned by one and only one tenant. Here I have assumed that a tenant must have at least one lease and if he wishes to extend the lease or sign another lease with the same housing agency in a different apartment he can have multiple leases.
- Each user (Employee) has one and only one assigned job type. Here I have assumed there is only a single employee specialized in a specific job type.
- A job type can appear in zero or many work order requests; Each work order request can be a part of a single job type.
- Each room may have several leases; There is one and only one lease for each room at a given time.
- An apartment has at least one room; A room belongs to only that specific apartment.
- An App User (Employee) may work on several work order requests.
- At least one employee works upon a work order request. Employee Work Order Assignment is an associative entity between the Employee and WorkOrder tables as they have a many to many relationships.
- There can be zero or multiple work order requests for each apartment; Each workorder request is associated with one and only one apartment.

Access Relationship Diagram:



Database Diagram – SQL Server:

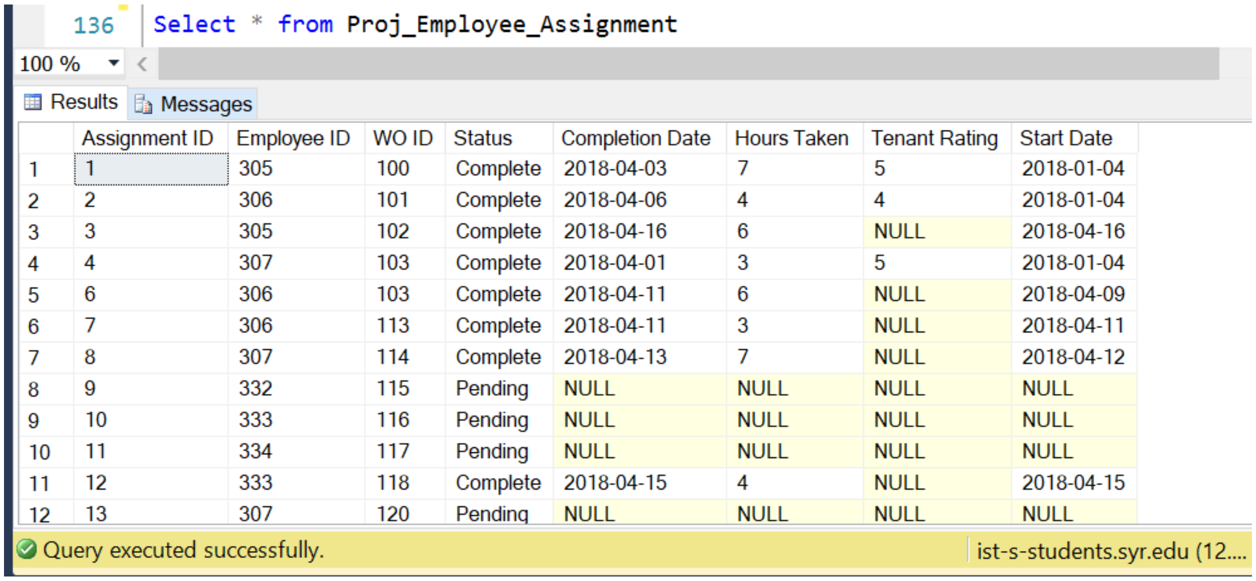


SQL Scripts for Creating and Inserting Data:

Table Creation Script:

```
-- Dropping Tables if they exist
Drop table Proj_Login;
Drop table Proj_Apartment;
Drop table Proj_AppUser;
Drop table Proj_JobType;
Drop table Proj_Employee_Assignment;
Drop table Proj_Lease;
Drop table Proj_Role;
Drop table Proj_Room;
Drop table Proj_WorkOrder_Request;

-- Creating Tables
-- Create table App User
CREATE TABLE [Proj_AppUser]
(
       [UserID] int NOT NULL,
       [FirstName] [varchar](20) NOT NULL,
       [LastName] [varchar](20) NOT NULL,
       [EmailID] [varchar](20) NOT NULL,
       [ContactNo] [varchar](10) NOT NULL,
       [Gender] [varchar](7) NOT NULL,
       [Job Type] int FOREIGN KEY REFERENCES Proj_JobType([Job ID]),
       Constraint AppUser_PK PRIMARY KEY(UserID),
       Constraint App_User_FK FOREIGN KEY(UserID) REFERENCES Proj_Login([Login ID])
) ;

-- Create table Role
CREATE TABLE [Proj_Role]
(
       [Role ID] int IDENTITY NOT NULL,
       [Role Name] varchar(10) NOT NULL,
       [Role Description] varchar(30) NULL,
       Constraint Role_PK PRIMARY KEY([Role ID])
);
-- Create table Job Type
Create Table [Job Type]
(
       [Job ID] int IDENTITY NOT NULL,
       [Job Type] varchar(20) NOT NULL
       Constraint Job_PK PRIMARY KEY([Job ID])
);

-- Create table Login
CREATE TABLE [Proj_Login]
(
       [Login ID] INT  IDENTITY(300,1) NOT NULL,
       [UserName] [varchar](20) NOT NULL,
       [Password] [varchar](20) NOT NULL,
       [Role ID] int NOT NULL,
       Constraint Login_PK PRIMARY KEY([Login ID]),
       Constraint Login_FK FOREIGN KEY ([Role ID]) REFERENCES [Proj_Role]([Role ID])
);
```

```
-- Create table Apartment
CREATE TABLE [Proj_Apartment]
(
	[Apartment No] varchar(6) NOT NULL,
	[Count Bedrooms] int NOT NULL,
	[Count Bathrooms] int Not NULL,
	[Garage] char(1) NOT NULL,
	[Laundry] char(1) NOT NULL,
	Constraint Apartment_PK PRIMARY KEY([Apartment No])
);

-- Create table Room
Create Table [Proj_Room]
(
	[Room_ID] int NOT NULL,
	[Apartment No] varchar(6) NOT NULL,
	[Furnished] char(1) NOT NULL,
	[Monthly Rent] float NOT NULL,
	[Room Dimension] varchar(10) NOT NULL,
	[Room Description] varchar(20) NULL
	Constraint Room_PK PRIMARY KEY([Room_ID]),
	Constraint Room_FK FOREIGN KEY([Apartment No]) REFERENCES
	Proj_Apartment([Apartment No])
);

-- Create table Lease
Create Table Proj_Lease
(
	[Lease ID]  int IDENTITY NOT NULL,
	[Tenant ID] int NOT NULL,
	[Room_ID] int NOT NULL,
	[Lease Date] Date NOT NULL,
	[Lease Start Date] Date NOT NULL,
	[Lease End Date] Date NOT NULL,
	[Security Deposit] float NOT NULL,
	Constraint Lease_PK PRIMARY KEY([Lease ID]),
	Constraint Lease_FK1 FOREIGN KEY ([Tenant ID]) REFERENCES Proj_AppUSer(UserID),
	Constraint Lease_FK2 FOREIGN KEY ([Room_ID]) REFERENCES Proj_Room([Room_ID])
);

-- Create table Work Order Request
Create table [Proj_WorkOrder_Request]
(
	[WO ID] int IDENTITY(100,1) NOT NULL,
	[Apartment No] varchar(6) NOT NULL,
	[Job Type] int FOREIGN KEY REFERENCES Proj_JobType([Job ID]) NOT NULL,
	[Priority] varchar(10) NOT NULL,
	[Request Date] Date NOT NULL,
	Constraint WorkOrder_PK PRIMARY KEY ([WO ID]),
	Constraint WorkOrder_FK FOREIGN KEY([Apartment No]) REFERENCES
	Proj_Apartment([Apartment No])
);

-- Create table Employee Assignment
Create table [Proj_Employee_Assignment]
(
        [Assignment ID] int IDENTITY NOT NULL,
        [Employee ID] int NOT NULL,
        [WO ID] int NOT NULL,
        [Status] varchar(10) NOT NULL,
        [Start Date] Date NULL,
        [Completion Date] Date NULL,
        [Hours Taken] int NULL,
        [Tenant Rating] int NULL,
        Constraint Assignment_PK PRIMARY KEY([Assignment ID]),
        Constraint Assignment_FK1 FOREIGN KEY([Employee ID]) REFERENCES
        Proj_AppUser(UserID),
        Constraint Assignment_FK2 FOREIGN KEY([WO ID]) REFERENCES
        [Proj_WorkOrder_Request]([WO ID])
);

-- Altering Employee Assignment to add default constraint
ALTER TABLE Proj_Employee_Assignment ADD CONSTRAINT DF_Status DEFAULT N'Pending' FOR
Status;


-- Insertions
-- Inserting Employee Assignment
Insert into Proj_Employee_Assignment([Employee ID],[WO ID],Status,[Completion
Date],[Hours Taken],[Tenant Rating])
values(305,100,'Complete','04/03/2018',7,5)
Insert into Proj_Employee_Assignment([Employee ID],[WO ID],Status,[Completion
Date],[Hours Taken],[Tenant Rating])
values(306,101,'Complete','04/06/2018',4,4)
Insert into Proj_Employee_Assignment([Employee ID],[WO ID],Status)
values(305,102,'Pending')
Insert into Proj_Employee_Assignment([Employee ID],[WO ID],Status,[Completion
Date],[Hours Taken],[Tenant Rating])
values(307,103,'Complete','04/01/2018',3,5)
Insert into Proj_Employee_Assignment([Employee ID],[WO ID]) values (306,103)

Select * from Proj_Employee_Assignment
```



```
Select * from Proj_Employee_Assignment
```

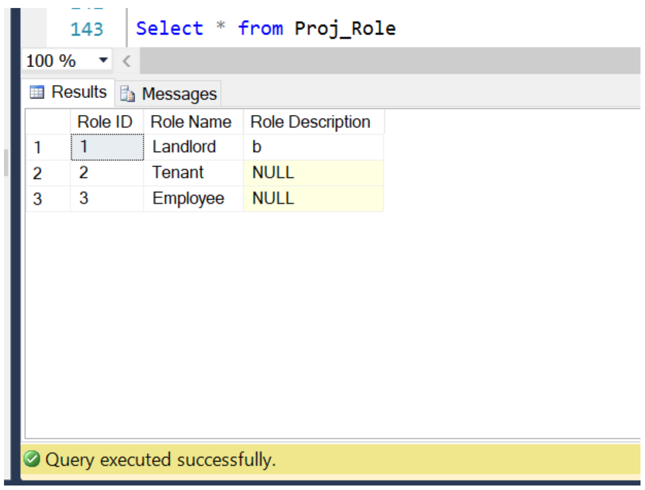
100 %

Results | Messages

	Assignment ID	Employee ID	WO ID	Status	Completion Date	Hours Taken	Tenant Rating	Start Date
1	1	305	100	Complete	2018-04-03	7	5	2018-01-04
2	2	306	101	Complete	2018-04-06	4	4	2018-01-04
3	3	305	102	Complete	2018-04-16	6	NULL	2018-04-16
4	4	307	103	Complete	2018-04-01	3	5	2018-01-04
5	6	306	103	Complete	2018-04-11	6	NULL	2018-04-09
6	7	306	113	Complete	2018-04-11	3	NULL	2018-04-11
7	8	307	114	Complete	2018-04-13	7	NULL	2018-04-12
8	9	332	115	Pending	NULL	NULL	NULL	NULL
9	10	333	116	Pending	NULL	NULL	NULL	NULL
10	11	334	117	Pending	NULL	NULL	NULL	NULL
11	12	333	118	Complete	2018-04-15	4	NULL	2018-04-15
12	13	307	120	Pending	NULL	NULL	NULL	NULL

Query executed successfully. ist-s-students.syr.edu (12....

```
--Inserting Role
Insert into Proj_Role([Role Name]) values('Landlord')
Insert into Proj_Role([Role Name]) values('Tenant');
Insert into Proj_Role([Role Name]) values('Employee')

Select * from Proj_Role
```

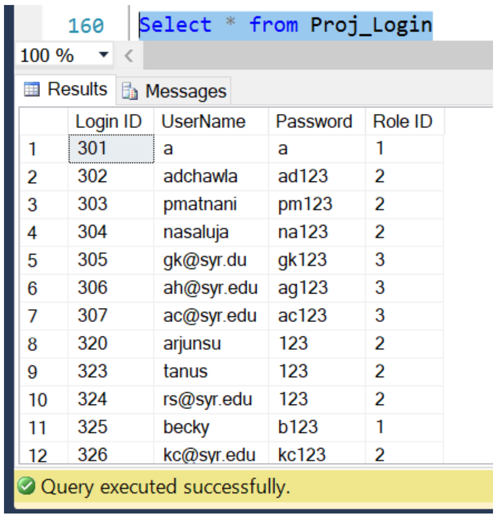


```
Select * from Proj_Role
```

	Role ID	Role Name	Role Description
1	1	Landlord	b
2	2	Tenant	NULL
3	3	Employee	NULL

Query executed successfully.

```
-- Inserting Login
Insert into Proj_Login(UserName,Password,[Role ID]) values ('adchawla','ad123',2)
Insert into Proj_Login(UserName,Password,[Role ID]) values ('becky','b123',1)
Insert into Proj_Login(UserName,Password,[Role ID]) values ('pmatnani','pm123',2)
Insert into Proj_Login(UserName,Password,[Role ID]) values ('nasaluja','na123',2)
Insert into Proj_Login(UserName,Password,[Role ID]) values ('gk@syr.du','gk123',3)
Insert into Proj_Login(UserName,Password,[Role ID]) values ('ah@syr.edu','ag123',3)
Insert into Proj_Login(UserName,Password,[Role ID]) values ('ac@syr.edu','ac123',3)
Insert into Proj_Login(UserName,Password,[Role ID]) values ('kc@syr.edu','kc123',2)
Insert into Proj_Login(UserName,Password,[Role ID]) values ('pc@syr.edu','pc123',2)
Insert into Proj_Login(UserName,Password,[Role ID]) values ('qc@syr.edu','qc123',2)
Insert into Proj_Login(UserName,Password,[Role ID]) values ('ack@syr.edu','ack123',3)
Insert into Proj_Login(UserName,Password,[Role ID]) values ('sks@syr.edu','sks123',3)
Insert into Proj_Login(UserName,Password,[Role ID]) values ('abc@syr.edu','abc123',3)

Select * from Proj_Login
```

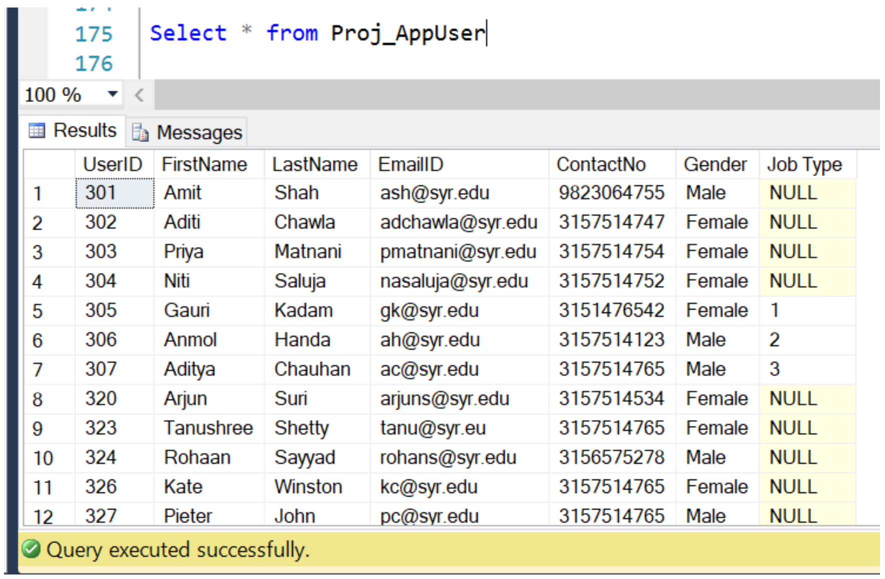


```
Select * from Proj_Login
```

	Login ID	UserName	Password	Role ID
1	301	a	a	1
2	302	adchawla	ad123	2
3	303	pmatnani	pm123	2
4	304	nasaluja	na123	2
5	305	gk@syr.du	gk123	3
6	306	ah@syr.edu	ag123	3
7	307	ac@syr.edu	ac123	3
8	320	arjunsu	123	2
9	323	tanus	123	2
10	324	rs@syr.edu	123	2
11	325	becky	b123	1
12	326	kc@syr.edu	kc123	2

Query executed successfully.

```
-- Inserting App User
Insert into Proj_AppUser
values(304,'Niti','Saluja','nasaluja@syr.edu','3157514752','Female');
Insert into Proj_AppUser
values(302,'Aditi','Chawla','adchawla@syr.edu','3157514747','Female');
Insert into Proj_AppUser
values(303,'Priya','Matnani','pmatnani@syr.edu','3157514754','Female');
Insert into Proj_AppUser values(305,'Gauri','Kadam','gk@syr.edu','3151476542','Female');
Insert into Proj_AppUser values(306,'Anmol','Handa','ah@syr.edu','3157514123','Male');
Insert into Proj_AppUser
values(332,'Anjali','Nair','ack@syr.edu','3157514123','Female',4);
Insert into Proj_AppUser
values(333,'Santosh','Shah','sks@syr.edu','3157514123','Female',5);
Insert into Proj_AppUser
values(334,'Amruta','Patil','abc@syr.edu','3157514123','Male',6);
Insert into Proj_AppUser(UserID,FirstName,LastName,EmailID,ContactNo,Gender)
values(326,'Kate','Winston','kc@syr.edu','3157514765','Female');
Insert into Proj_AppUser(UserID,FirstName,LastName,EmailID,ContactNo,Gender)
values(327,'Pieter','John','pc@syr.edu','3157514765','Male');
Insert into Proj_AppUser(UserID,FirstName,LastName,EmailID,ContactNo,Gender)
values(328,'Qunfang','Wu','qc@syr.edu','3157514765','Female');

Select * from Proj_AppUser
```

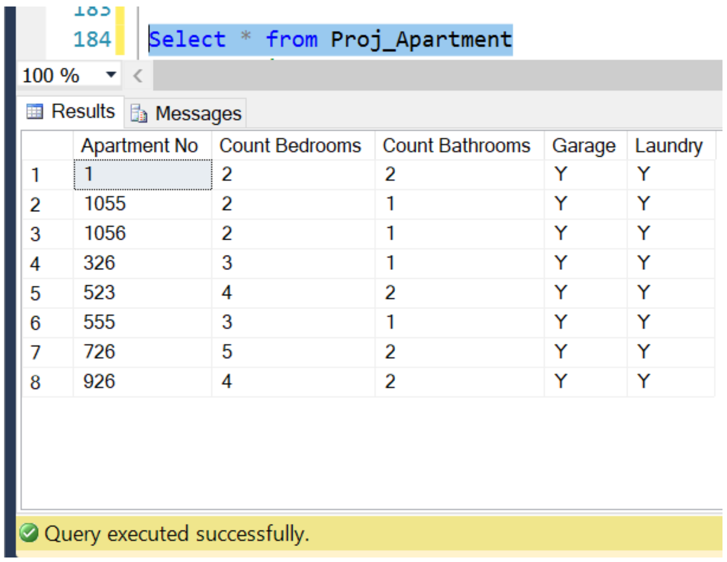


```
Select * from Proj_AppUser
```

	UserID	FirstName	LastName	EmailID	ContactNo	Gender	Job Type
1	301	Amit	Shah	ash@syr.edu	9823064755	Male	NULL
2	302	Aditi	Chawla	adchawla@syr.edu	3157514747	Female	NULL
3	303	Priya	Matnani	pmatnani@syr.edu	3157514754	Female	NULL
4	304	Niti	Saluja	nasaluja@syr.edu	3157514752	Female	NULL
5	305	Gauri	Kadam	gk@syr.edu	3151476542	Female	1
6	306	Anmol	Handa	ah@syr.edu	3157514123	Male	2
7	307	Aditya	Chauhan	ac@syr.edu	3157514765	Male	3
8	320	Arjun	Suri	arjuns@syr.edu	3157514534	Female	NULL
9	323	Tanushree	Shetty	tanu@syr.eu	3157514765	Female	NULL
10	324	Rohaan	Sayyad	rohans@syr.edu	3156575278	Male	NULL
11	326	Kate	Winston	kc@syr.edu	3157514765	Female	NULL
12	327	Pieter	John	pc@syr.edu	3157514765	Male	NULL

Query executed successfully.

```
--Inserting Apartment
Insert into Proj_Apartment values ('326',3,1,'Y','Y')
Insert into Proj_Apartment values ('523',4,2,'Y','Y')
Insert into Proj_Apartment values ('555',3,1,'Y','Y')
Insert into Proj_Apartment values ('726',5,2,'Y','Y')
Insert into Proj_Apartment values ('926',4,2,'Y','Y')

Select * from Proj_Apartment;
```



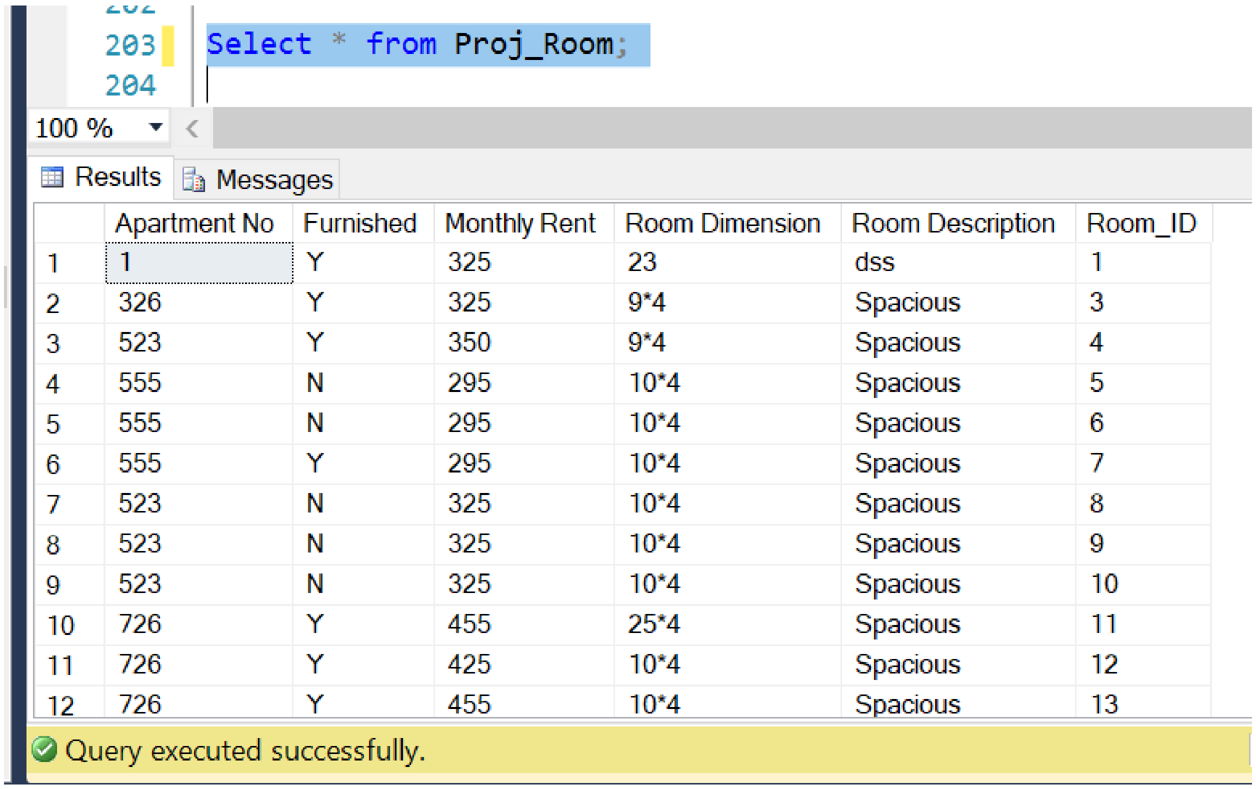
```
Select * from Proj_Apartment
```

	Apartment No	Count Bedrooms	Count Bathrooms	Garage	Laundry
1	1	2	2	Y	Y
2	1055	2	1	Y	Y
3	1056	2	1	Y	Y
4	326	3	1	Y	Y
5	523	4	2	Y	Y
6	555	3	1	Y	Y
7	726	5	2	Y	Y
8	926	4	2	Y	Y

Query executed successfully.

```
-- Inserting Room
Insert into Proj_Room([Apartment No],[Furnished],[Monthly Rent],[Room Dimension],[Room
Description])values ('326','Y',325,'9*4','Spacious')
Insert into Proj_Room([Apartment No],[Furnished],[Monthly Rent],[Room Dimension],[Room
Description])values ('523','Y',350,'9*4','Spacious')
Insert into Proj_Room([Apartment No],[Furnished],[Monthly Rent],[Room Dimension],[Room
Description])values ('555','N',295,'10*4','Spacious')
Insert into Proj_Room([Apartment No],[Furnished],[Monthly Rent],[Room Dimension],[Room
Description])values ('555','N',295,'10*4','Spacious')
```

```
Insert into Proj_Room([Apartment No],[Furnished],[Monthly Rent],[Room Dimension],[Room
Description])values ('555','Y',295,'10*4','Spacious')
Insert into Proj_Room([Apartment No],[Furnished],[Monthly Rent],[Room Dimension],[Room
Description])values ('523','N',325,'10*4','Spacious')
Insert into Proj_Room([Apartment No],[Furnished],[Monthly Rent],[Room Dimension],[Room
Description])values ('523','N',325,'10*4','Spacious')
Insert into Proj_Room([Apartment No],[Furnished],[Monthly Rent],[Room Dimension],[Room
Description])values ('523','N',325,'10*4','Spacious')
Insert into Proj_Room([Apartment No],[Furnished],[Monthly Rent],[Room Dimension],[Room
Description])values ('726','Y',455,'25*4','Spacious')
Insert into Proj_Room([Apartment No],[Furnished],[Monthly Rent],[Room Dimension],[Room
Description])values ('726','Y',425,'10*4','Spacious')
Insert into Proj_Room([Apartment No],[Furnished],[Monthly Rent],[Room Dimension],[Room
Description])values ('726','Y',455,'10*4','Spacious')
Insert into Proj_Room([Apartment No],[Furnished],[Monthly Rent],[Room Dimension],[Room
Description])values ('726','Y',350,'10*4','Spacious')
Insert into Proj_Room([Apartment No],[Furnished],[Monthly Rent],[Room Dimension],[Room
Description])values ('726','Y',450,'10*4','Spacious')
Insert into Proj_Room([Apartment No],[Furnished],[Monthly Rent],[Room Dimension],[Room
Description])values ('926','N',350,'10*4','Spacious')
Insert into Proj_Room([Apartment No],[Furnished],[Monthly Rent],[Room Dimension],[Room
Description])values ('926','N',350,'10*4','Spacious')
Insert into Proj_Room([Apartment No],[Furnished],[Monthly Rent],[Room Dimension],[Room
Description])values ('926','N', 370,'10*4','Spacious')

Select * from Proj_Room
```



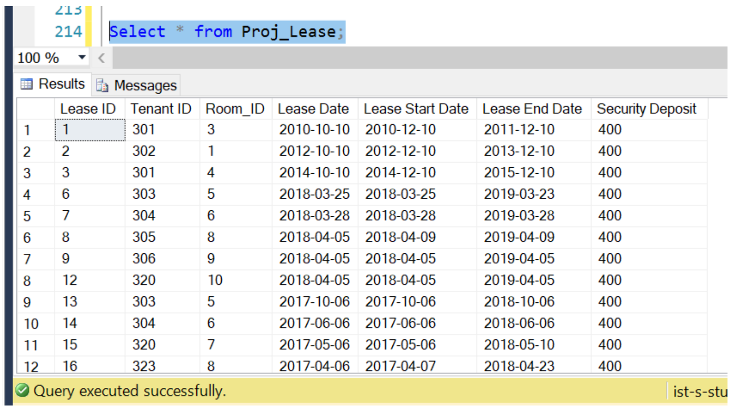
```
Select * from Proj_Room;
```

	Apartment No	Furnished	Monthly Rent	Room Dimension	Room Description	Room_ID
1	1	Y	325	23	dss	1
2	326	Y	325	9*4	Spacious	3
3	523	Y	350	9*4	Spacious	4
4	555	N	295	10*4	Spacious	5
5	555	N	295	10*4	Spacious	6
6	555	Y	295	10*4	Spacious	7
7	523	N	325	10*4	Spacious	8
8	523	N	325	10*4	Spacious	9
9	523	N	325	10*4	Spacious	10
10	726	Y	455	25*4	Spacious	11
11	726	Y	425	10*4	Spacious	12
12	726	Y	455	10*4	Spacious	13

Query executed successfully.

```
-- Inserting Lease
Insert into Proj_Lease ( [Tenant ID],Room_ID,[Lease Date],[Lease Start Date],[Lease End
Date],[Security Deposit]) values (301,3,'10/10/2010','12/10/2010','12/10/2011',400)
```

```
Insert into Proj_Lease ( [Tenant ID],Room_ID,[Lease Date],[Lease Start Date],[Lease End
Date],[Security Deposit]) values (302,1,'10/10/2012','12/10/2012','12/10/2013',400)
Insert into Proj_Lease ( [Tenant ID],Room_ID,[Lease Date],[Lease Start Date],[Lease End
Date],[Security Deposit]) values (301,4,'10/10/2014','12/10/2014','12/10/2015',400)
Insert into Proj_Lease ( [Tenant ID],Room_ID,[Lease Date],[Lease Start Date],[Lease End
Date],[Security Deposit]) values (303,5,'10/06/2017','10/06/2017','10/06/2018',400)
Insert into Proj_Lease ( [Tenant ID],Room_ID,[Lease Date],[Lease Start Date],[Lease End
Date],[Security Deposit]) values (304,6,'06/06/2017','06/06/2017','06/06/2018',400)
Insert into Proj_Lease ( [Tenant ID],Room_ID,[Lease Date],[Lease Start Date],[Lease End
Date],[Security Deposit]) values (320,7,'05/06/2017','05/06/2017','05/10/2018',400)
Insert into Proj_Lease ( [Tenant ID],Room_ID,[Lease Date],[Lease Start Date],[Lease End
Date],[Security Deposit]) values (323,8,'04/06/2017','04/07/2017','04/23/2018',400)
```

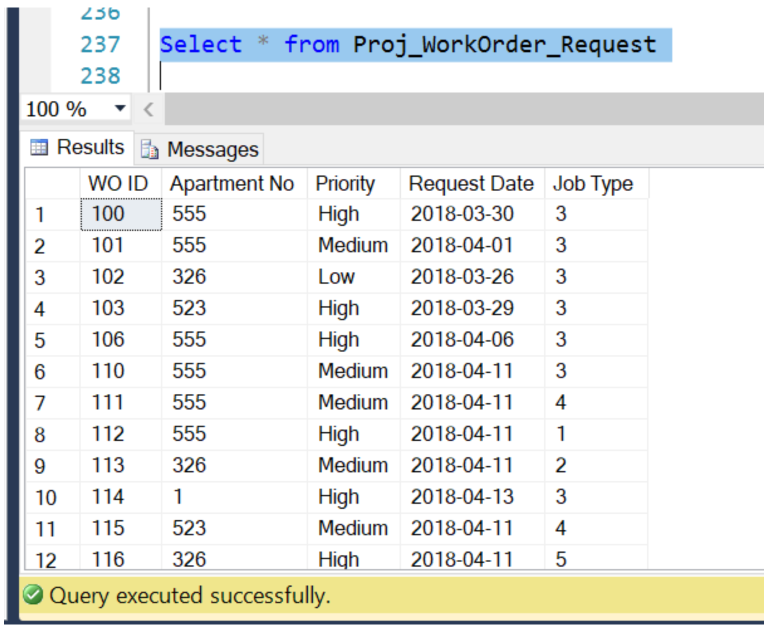


```
Select * from Proj_Lease;
```

	Lease ID	Tenant ID	Room_ID	Lease Date	Lease Start Date	Lease End Date	Security Deposit
1	1	301	3	2010-10-10	2010-12-10	2011-12-10	400
2	2	302	1	2012-10-10	2012-12-10	2013-12-10	400
3	3	301	4	2014-10-10	2014-12-10	2015-12-10	400
4	6	303	5	2018-03-25	2018-03-25	2019-03-23	400
5	7	304	6	2018-03-28	2018-03-28	2019-03-28	400
6	8	305	8	2018-04-05	2018-04-09	2019-04-09	400
7	9	306	9	2018-04-05	2018-04-05	2019-04-05	400
8	12	320	10	2018-04-05	2018-04-05	2019-04-05	400
9	13	303	5	2017-10-06	2017-10-06	2018-10-06	400
10	14	304	6	2017-06-06	2017-06-06	2018-06-06	400
11	15	320	7	2017-05-06	2017-05-06	2018-05-10	400
12	16	323	8	2017-04-06	2017-04-07	2018-04-23	400

Query executed successfully.

```
-- Inserting Work Order Requests

Insert into [Proj_WorkOrder_Request]([Apartment No],[Job Type],Priority,[Request Date])
values(555,'Cleaning','High','03/30/2018')
Insert into [Proj_WorkOrder_Request]([Apartment No],[Job Type],Priority,[Request Date])
values(555,'Bed Bugs','Medium','04/01/2018')
Insert into [Proj_WorkOrder_Request]([Apartment No],[Job Type],Priority,[Request Date])
values(326,'Heater','Low','03/26/2018')
Insert into [Proj_WorkOrder_Request]([Apartment No],[Job Type],Priority,[Request Date])
values(523,'Cleaning','High','03/29/2018')
Insert into [Proj_WorkOrder_Request]([Apartment No],[Job Type],Priority,[Request Date])
values(555,'Sewage Treatment','Medium','04/11/2018')
Insert into [Proj_WorkOrder_Request]([Apartment No],[Job Type],Priority,[Request Date])
values(326,2,'Medium','04/11/2018')
Insert into [Proj_WorkOrder_Request]([Apartment No],[Job Type],Priority,[Request Date])
values(523,4,'Medium','04/11/2018')
Insert into [Proj_WorkOrder_Request]([Apartment No],[Job Type],Priority,[Request Date])
values(326,5,'High','04/11/2018')
Insert into [Proj_WorkOrder_Request]([Apartment No],[Job Type],Priority,[Request Date])
values(326,6,'High','04/11/2018')

Select * from Proj_WorkOrder_Request
```

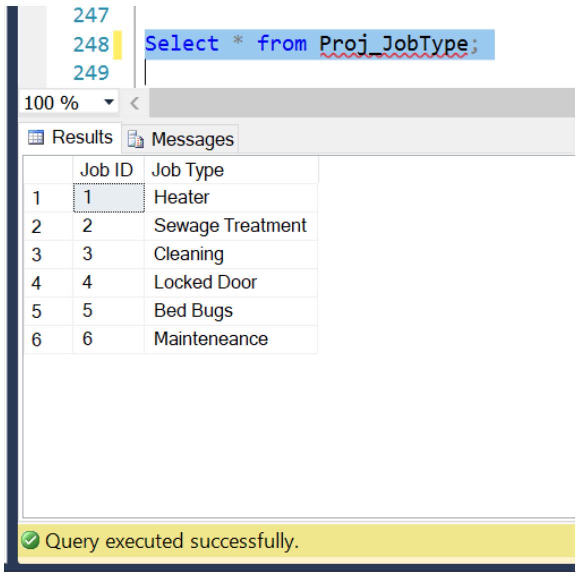


```
Select * from Proj_WorkOrder_Request
```

	WO ID	Apartment No	Priority	Request Date	Job Type
1	100	555	High	2018-03-30	3
2	101	555	Medium	2018-04-01	3
3	102	326	Low	2018-03-26	3
4	103	523	High	2018-03-29	3
5	106	555	High	2018-04-06	3
6	110	555	Medium	2018-04-11	3
7	111	555	Medium	2018-04-11	4
8	112	555	High	2018-04-11	1
9	113	326	Medium	2018-04-11	2
10	114	1	High	2018-04-13	3
11	115	523	Medium	2018-04-11	4
12	116	326	High	2018-04-11	5

Query executed successfully.

```
--- Inserting Job Type
Insert into [Job Type] ([Job Type]) values('Heater')
Insert into [Job Type] ([Job Type]) values('Sewage Treatment')
Insert into [Job Type] ([Job Type]) values('Cleaning')
Insert into [Job Type] ([Job Type]) values('Locked Door')
Insert into [Job Type] ([Job Type]) values('Bed Bugs')
Insert into [Job Type] ([Job Type]) values('Mainteneance')

Select * from Proj_JobType
```



```
Select * from Proj_JobType;
```

	Job ID	Job Type
1	1	Heater
2	2	Sewage Treatment
3	3	Cleaning
4	4	Locked Door
5	5	Bed Bugs
6	6	Mainteneance

Query executed successfully.

Major Data Questions:

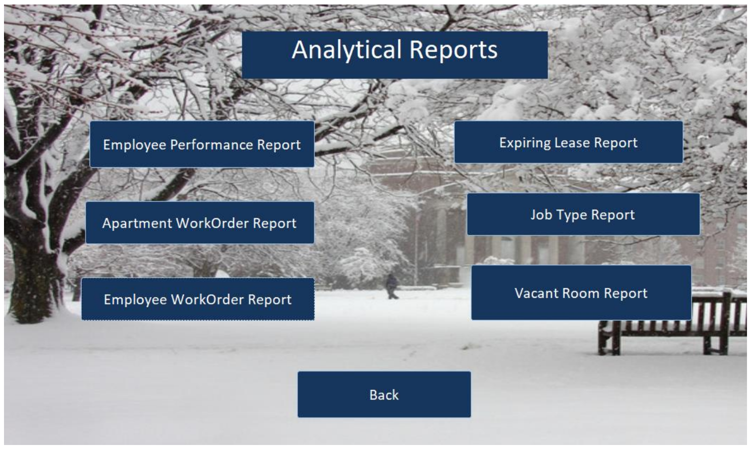
Syracuse Apartment Renting system manages all the activities of the property owners centrally. It also caters to the needs of the tenants by streamlining the complex process of logging work order requests. There are three users of the system:

i) Landlord/Rental Manager
ii) Tenant
iii) Employee

The below list highlights upon some of the data questions which are answered by the users of the proposed system:

i)<u>The Landlord queries the database for:</u>

<u>Reports:</u>

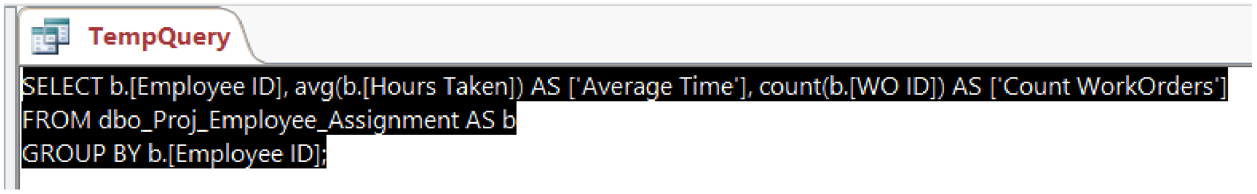

Analytical Reports
Employee Performance Report
Expiring Lease Report
Apartment WorkOrder Report
Job Type Report
Employee WorkOrder Report
Vacant Room Report
Back

- **Best Performing Employee:**
  This report will help the property owners to select the best performing employee by analyzing the performance of each employee, which is measured using the average time and the count of work orders worked upon by the employees. This information is retrieved from the association table of Employee and WorkOrder

  Query Used:

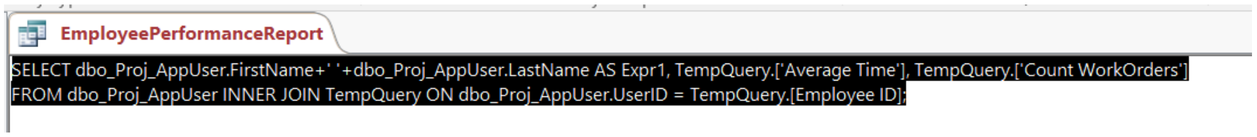
  TempQuery (Sub Query)

  ```
  SELECT b.[Employee ID], avg(b.[Hours Taken]) AS ['Average Time'], count(b.[WOID]) AS ['Count WorkOrders'] FROM dbo_Proj_Employee_Assignment AS b GROUP BY b.[Employee ID];
  ```



```
SELECT b.[Employee ID], avg(b.[Hours Taken]) AS ['Average Time'], count(b.[WO ID]) AS ['Count WorkOrders']
FROM dbo_Proj_Employee_Assignment AS b
GROUP BY b.[Employee ID];
```

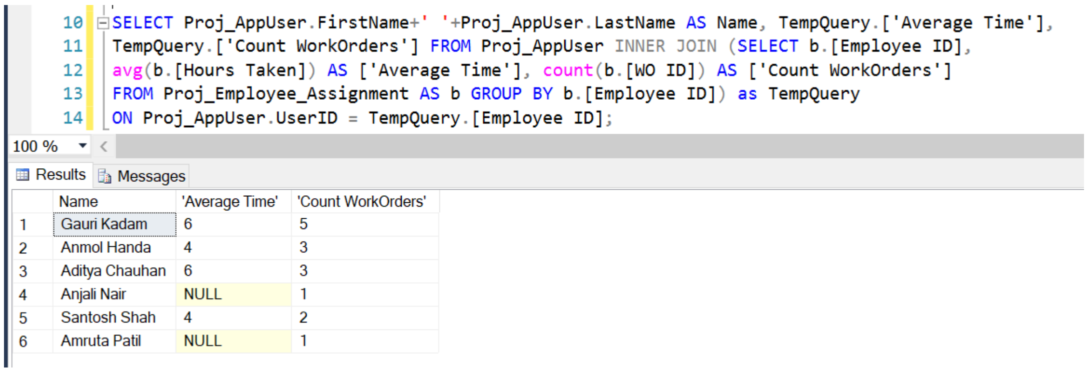
SELECT dbo_Proj_AppUser.FirstName+' '+dbo_Proj_AppUser.LastName AS Expr1, TempQuery.['Average Time'], TempQuery.['Count WorkOrders'] FROM dbo_Proj_AppUser INNER JOIN TempQuery ON dbo_Proj_AppUser.UserID = TempQuery.[Employee ID];



```
SELECT dbo_Proj_AppUser.FirstName+' '+dbo_Proj_AppUser.LastName AS Expr1, TempQuery.['Average Time'], TempQuery.['Count WorkOrders']
FROM dbo_Proj_AppUser INNER JOIN TempQuery ON dbo_Proj_AppUser.UserID = TempQuery.[Employee ID];
```

SQL View:

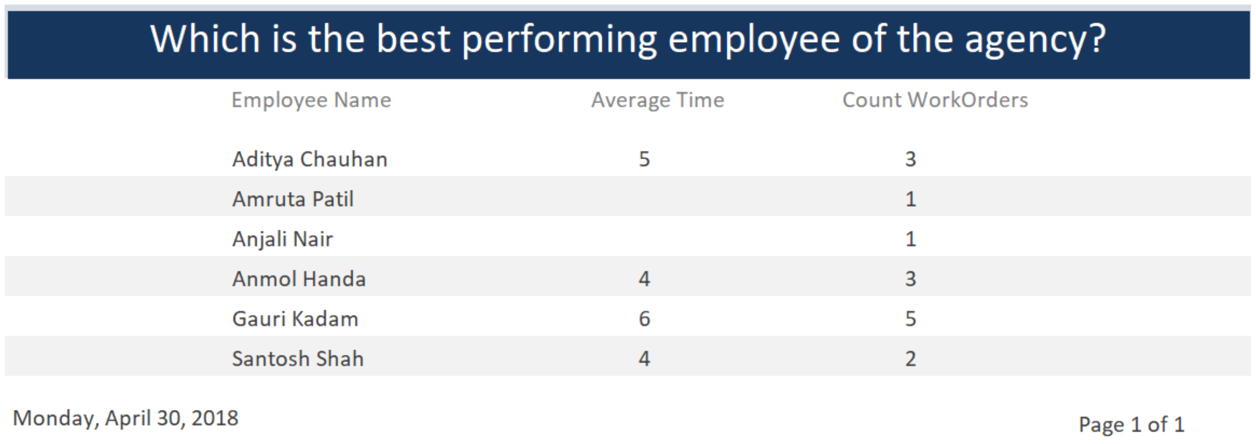
```
SELECT Proj_AppUser.FirstName+' '+Proj_AppUser.LastName AS Name,
TempQuery.['Average Time'],
TempQuery.['Count WorkOrders'] FROM Proj_AppUser INNER JOIN (SELECT b.[Employee
ID],
avg(b.[Hours Taken]) AS ['Average Time'], count(b.[WO ID]) AS ['Count WorkOrders']
FROM Proj_Employee_Assignment AS b GROUP BY b.[Employee ID]) as TempQuery
ON Proj_AppUser.UserID = TempQuery.[Employee ID];
```



```
SELECT Proj_AppUser.FirstName+' '+Proj_AppUser.LastName AS Name, TempQuery.['Average Time'],
TempQuery.['Count WorkOrders'] FROM Proj_AppUser INNER JOIN (SELECT b.[Employee ID],
avg(b.[Hours Taken]) AS ['Average Time'], count(b.[WO ID]) AS ['Count WorkOrders']
FROM Proj_Employee_Assignment AS b GROUP BY b.[Employee ID]) as TempQuery
ON Proj_AppUser.UserID = TempQuery.[Employee ID];
```

	Name	'Average Time'	'Count WorkOrders'
1	Gauri Kadam	6	5
2	Anmol Handa	4	3
3	Aditya Chauhan	6	3
4	Anjali Nair	NULL	1
5	Santosh Shah	4	2
6	Amruta Patil	NULL	1

Access Report View:



Which is the best performing employee of the agency?

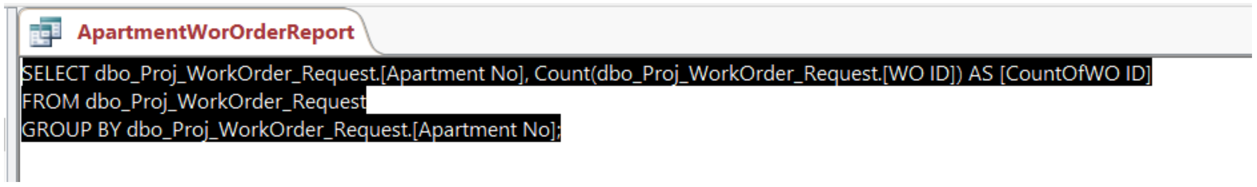
Employee Name	Average Time	Count WorkOrders
Aditya Chauhan	5	3
Amruta Patil		1
Anjali Nair		1
Anmol Handa	4	3
Gauri Kadam	6	5
Santosh Shah	4	2

Monday, April 30, 2018 Page 1 of 1

- **Apartment WorkOrder Report:**
  This report will help in analyzing the number of work orders requested for each apartment. This information will in turn help us to short list the apartments with maximum number of work orders for renovation.

  Query Used:

  SELECT dbo_Proj_WorkOrder_Request.[Apartment No],
  Count(dbo_Proj_WorkOrder_Request.[WO ID]) AS [CountOfWO ID]
  FROM dbo_Proj_WorkOrder_Request
  GROUP BY dbo_Proj_WorkOrder_Request.[Apartment No];

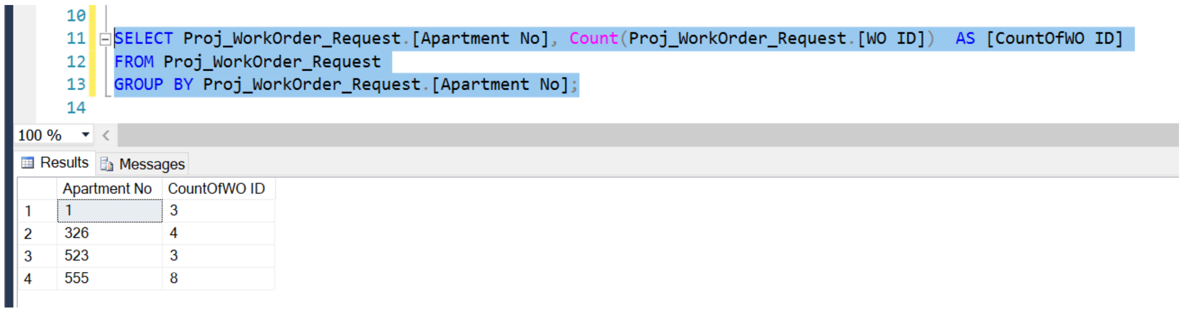


ApartmentWorOrderReport

```
SELECT dbo_Proj_WorkOrder_Request.[Apartment No], Count(dbo_Proj_WorkOrder_Request.[WO ID]) AS [CountOfWO ID]
FROM dbo_Proj_WorkOrder_Request
GROUP BY dbo_Proj_WorkOrder_Request.[Apartment No];
```

SQL View:

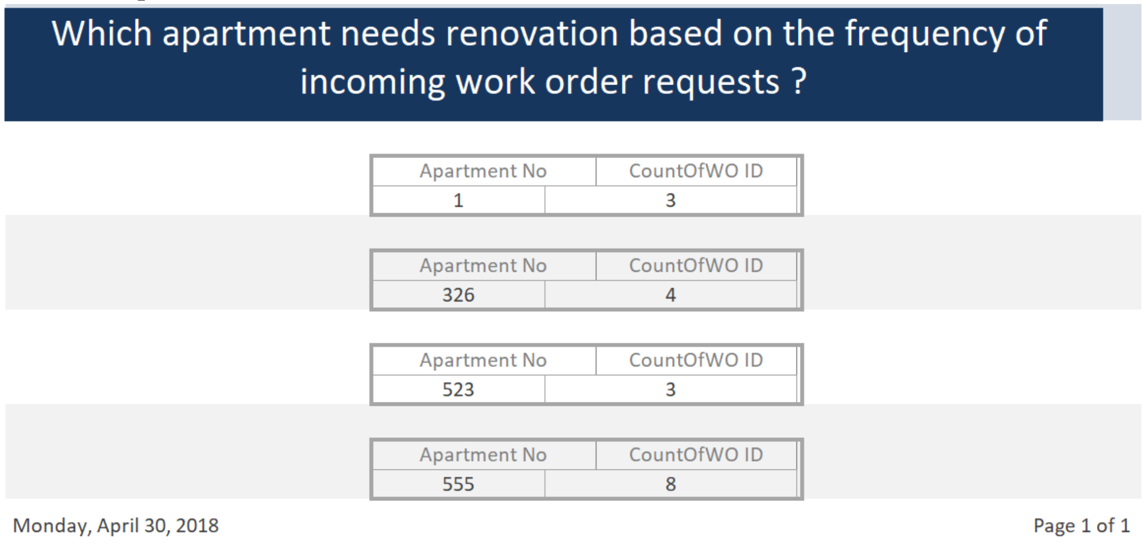
```
SELECT Proj_WorkOrder_Request.[Apartment No], Count(Proj_WorkOrder_Request.[WO
ID])  AS [CountOfWO ID]
FROM Proj_WorkOrder_Request
GROUP BY Proj_WorkOrder_Request.[Apartment No];
```



```
SELECT Proj_WorkOrder_Request.[Apartment No], Count(Proj_WorkOrder_Request.[WO ID])  AS [CountOfWO ID]
FROM Proj_WorkOrder_Request
GROUP BY Proj_WorkOrder_Request.[Apartment No];
```

	Apartment No	CountOfWO ID
1	1	3
2	326	4
3	523	3
4	555	8

Access Report View:



Which apartment needs renovation based on the frequency of incoming work order requests ?

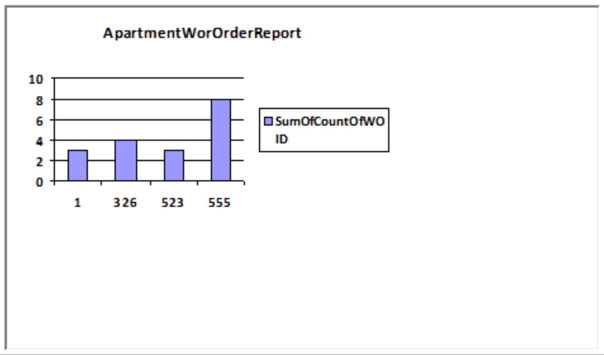
Apartment No	CountOfWO ID
1	3

Apartment No	CountOfWO ID
326	4

Apartment No	CountOfWO ID
523	3

Apartment No	CountOfWO ID
555	8

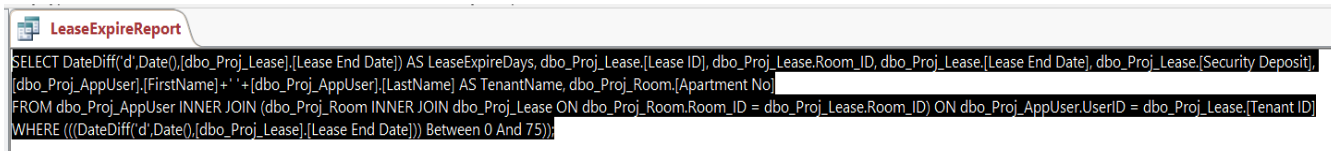
Monday, April 30, 2018 Page 1 of 1



- **Expiring Lease Report:**
This report will be useful for deriving information about the leases expiring in the next75 days so that the property owners can plan for their repair works and post advertisements for next year’s rental. This would help them to plan their further steps systematically.

  Query Used:

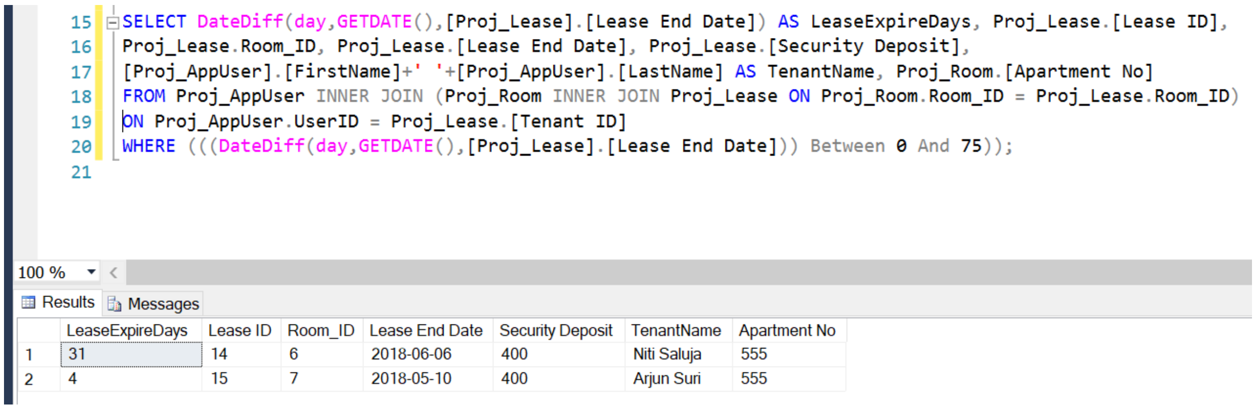
SELECT DateDiff('d',Date(),[dbo_Proj_Lease].[Lease End Date]) AS LeaseExpireDays, dbo_Proj_Lease.[Lease ID], dbo_Proj_Lease.Room_ID, dbo_Proj_Lease.[Lease End Date], dbo_Proj_Lease.[Security Deposit], [dbo_Proj_AppUser].[FirstName]+' '+[dbo_Proj_AppUser].[LastName] AS TenantName, dbo_Proj_Room.[Apartment No]
FROM dbo_Proj_AppUser INNER JOIN (dbo_Proj_Room INNER JOIN dbo_Proj_Lease ON dbo_Proj_Room.Room_ID = dbo_Proj_Lease.Room_ID) ON dbo_Proj_AppUser.UserID = dbo_Proj_Lease.[Tenant ID]
WHERE (((DateDiff('d',Date(),[dbo_Proj_Lease].[Lease End Date])) Between 0 And 75));


LeaseExpireReport

```
SELECT DateDiff('d',Date(),[dbo_Proj_Lease].[Lease End Date]) AS LeaseExpireDays, dbo_Proj_Lease.[Lease ID], dbo_Proj_Lease.Room_ID, dbo_Proj_Lease.[Lease End Date], dbo_Proj_Lease.[Security Deposit],
[dbo_Proj_AppUser].[FirstName]+' '+[dbo_Proj_AppUser].[LastName] AS TenantName, dbo_Proj_Room.[Apartment No]
FROM dbo_Proj_AppUser INNER JOIN (dbo_Proj_Room INNER JOIN dbo_Proj_Lease ON dbo_Proj_Room.Room_ID = dbo_Proj_Lease.Room_ID) ON dbo_Proj_AppUser.UserID = dbo_Proj_Lease.[Tenant ID]
WHERE (((DateDiff('d',Date(),[dbo_Proj_Lease].[Lease End Date])) Between 0 And 75));
```

SQL View:

```
SELECT DateDiff(day,GETDATE(),[Proj_Lease].[Lease End Date]) AS LeaseExpireDays,
Proj_Lease.[Lease ID],
Proj_Lease.Room_ID, Proj_Lease.[Lease End Date], Proj_Lease.[Security Deposit],
[Proj_AppUser].[FirstName]+' '+[Proj_AppUser].[LastName] AS TenantName,
Proj_Room.[Apartment No]
FROM Proj_AppUser INNER JOIN (Proj_Room INNER JOIN Proj_Lease ON Proj_Room.Room_ID
= Proj_Lease.Room_ID)
ON Proj_AppUser.UserID = Proj_Lease.[Tenant ID]
WHERE (((DateDiff(day,GETDATE(),[Proj_Lease].[Lease End Date])) Between 0 And
75));
```



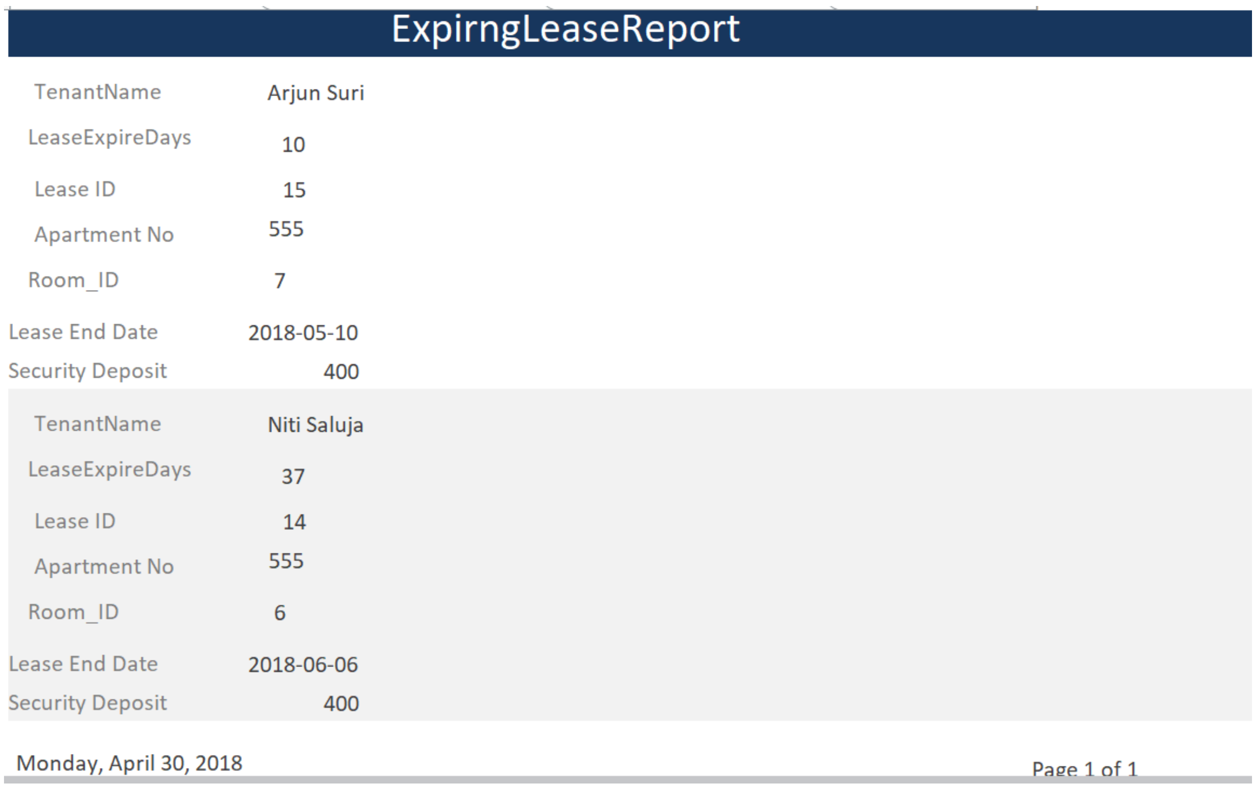
```
SELECT DateDiff(day,GETDATE(),[Proj_Lease].[Lease End Date]) AS LeaseExpireDays, Proj_Lease.[Lease ID],
Proj_Lease.Room_ID, Proj_Lease.[Lease End Date], Proj_Lease.[Security Deposit],
[Proj_AppUser].[FirstName]+' '+[Proj_AppUser].[LastName] AS TenantName, Proj_Room.[Apartment No]
FROM Proj_AppUser INNER JOIN (Proj_Room INNER JOIN Proj_Lease ON Proj_Room.Room_ID = Proj_Lease.Room_ID)
ON Proj_AppUser.UserID = Proj_Lease.[Tenant ID]
WHERE (((DateDiff(day,GETDATE(),[Proj_Lease].[Lease End Date])) Between 0 And 75));
```

100 %

Results | Messages

	LeaseExpireDays	Lease ID	Room_ID	Lease End Date	Security Deposit	TenantName	Apartment No
1	31	14	6	2018-06-06	400	Niti Saluja	555
2	4	15	7	2018-05-10	400	Arjun Suri	555

Access Report View:



ExpirngLeaseReport

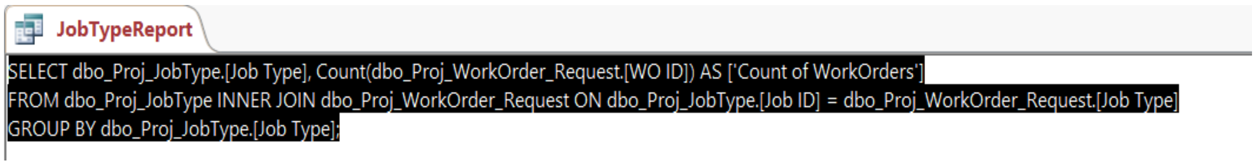
TenantName	Arjun Suri
LeaseExpireDays	10
Lease ID	15
Apartment No	555
Room_ID	7
Lease End Date	2018-05-10
Security Deposit	400
TenantName	Niti Saluja
LeaseExpireDays	37
Lease ID	14
Apartment No	555
Room_ID	6
Lease End Date	2018-06-06
Security Deposit	400

Monday, April 30, 2018 Page 1 of 1

- **Job Type Report:**
  This report would provide a single view for the number of work orders requested for each of the job type. This information would help the property owners to hire more employees of a specific job type if the count demands so.

  Query Used:

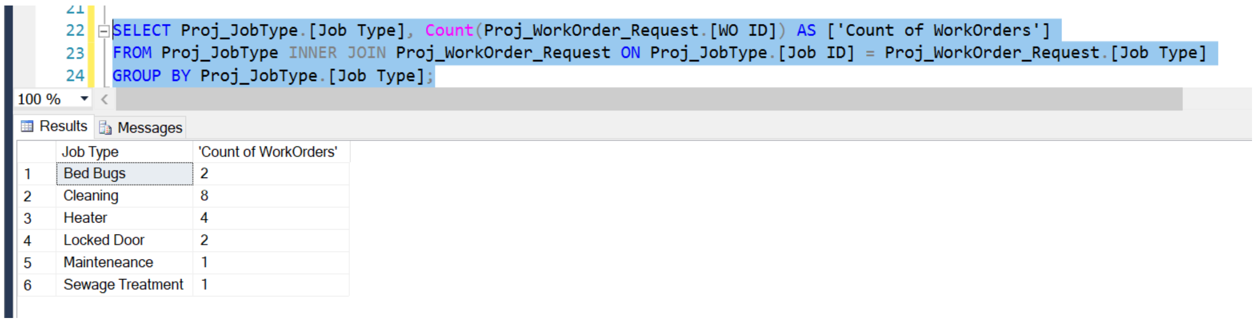
  SELECT dbo_Proj_JobType.[Job Type], Count(dbo_Proj_WorkOrder_Request.[WO ID]) AS ['Count of WorkOrders']
  FROM dbo_Proj_JobType INNER JOIN dbo_Proj_WorkOrder_Request ON dbo_Proj_JobType.[Job ID] = dbo_Proj_WorkOrder_Request.[Job Type]
  GROUP BY dbo_Proj_JobType.[Job Type];



```
SELECT dbo_Proj_JobType.[Job Type], Count(dbo_Proj_WorkOrder_Request.[WO ID]) AS ['Count of WorkOrders']
FROM dbo_Proj_JobType INNER JOIN dbo_Proj_WorkOrder_Request ON dbo_Proj_JobType.[Job ID] = dbo_Proj_WorkOrder_Request.[Job Type]
GROUP BY dbo_Proj_JobType.[Job Type];
```

SQL View:

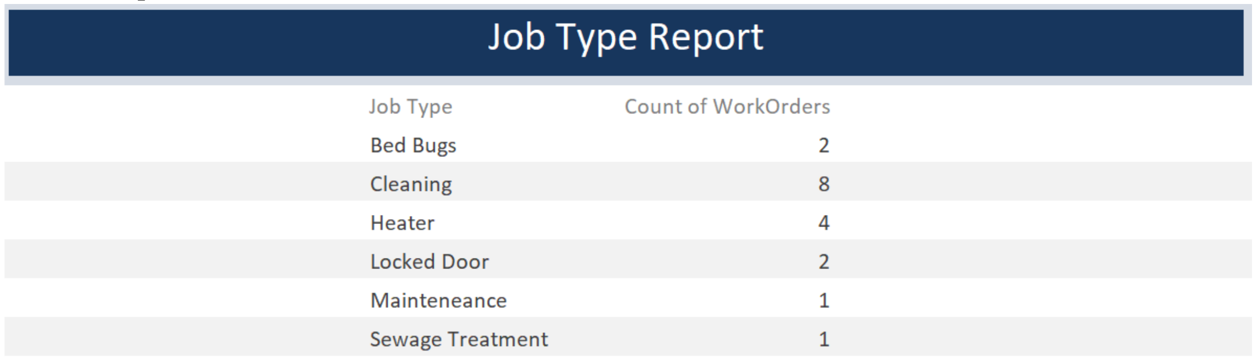
```
SELECT Proj_JobType.[Job Type], Count(Proj_WorkOrder_Request.[WO ID]) AS ['Count
of WorkOrders']
FROM Proj_JobType INNER JOIN Proj_WorkOrder_Request ON Proj_JobType.[Job ID] =
Proj_WorkOrder_Request.[Job Type]
GROUP BY Proj_JobType.[Job Type];
```



```
SELECT Proj_JobType.[Job Type], Count(Proj_WorkOrder_Request.[WO ID]) AS ['Count of WorkOrders']
FROM Proj_JobType INNER JOIN Proj_WorkOrder_Request ON Proj_JobType.[Job ID] = Proj_WorkOrder_Request.[Job Type]
GROUP BY Proj_JobType.[Job Type];
```

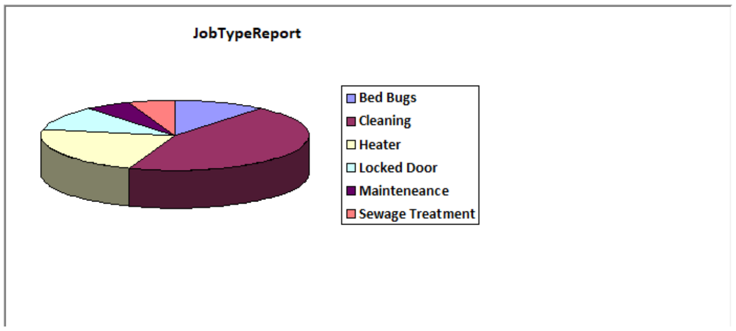
	Job Type	'Count of WorkOrders'
1	Bed Bugs	2
2	Cleaning	8
3	Heater	4
4	Locked Door	2
5	Mainteneance	1
6	Sewage Treatment	1

Access Report View:



Job Type Report

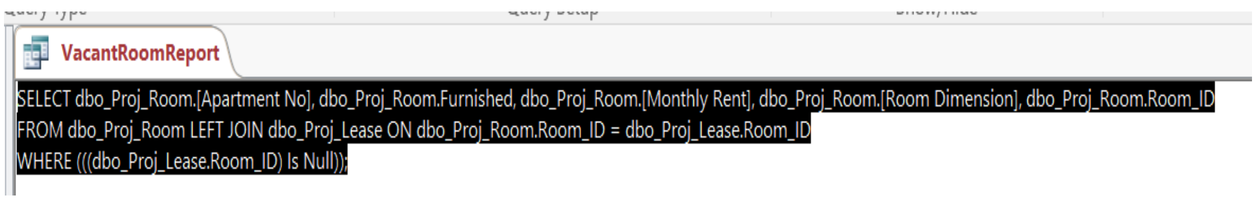
Job Type	Count of WorkOrders
Bed Bugs	2
Cleaning	8
Heater	4
Locked Door	2
Mainteneance	1
Sewage Treatment	1



- **Vacant Room Report:**
  This information will help in analyzing all the significant details about the vacant room so that various advertisements could be posted to rent them.

Query Used:

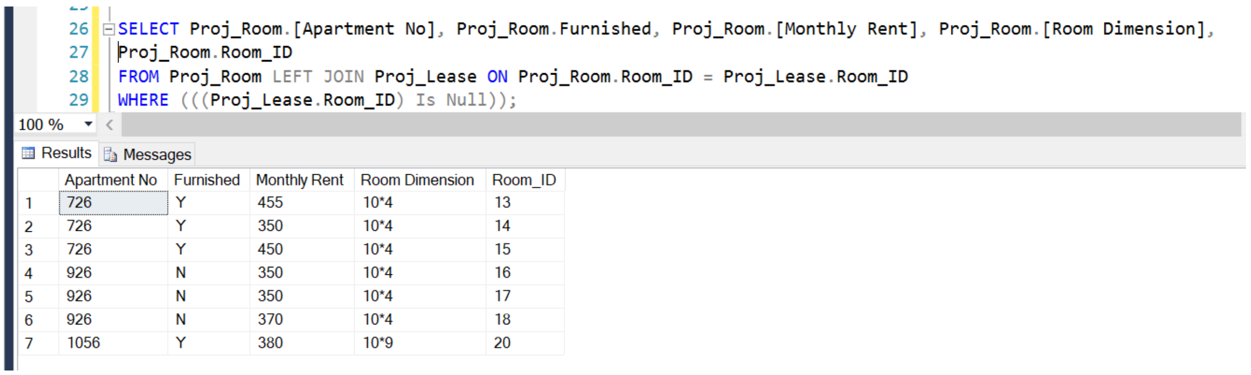
SELECT dbo_Proj_Room.[Apartment No], dbo_Proj_Room.Furnished, dbo_Proj_Room.[Monthly Rent], dbo_Proj_Room.[Room Dimension], dbo_Proj_Room.Room_ID
FROM dbo_Proj_Room LEFT JOIN dbo_Proj_Lease ON dbo_Proj_Room.Room_ID = dbo_Proj_Lease.Room_ID
WHERE (((dbo_Proj_Lease.Room_ID) Is Null));



```
SELECT dbo_Proj_Room.[Apartment No], dbo_Proj_Room.Furnished, dbo_Proj_Room.[Monthly Rent], dbo_Proj_Room.[Room Dimension], dbo_Proj_Room.Room_ID
FROM dbo_Proj_Room LEFT JOIN dbo_Proj_Lease ON dbo_Proj_Room.Room_ID = dbo_Proj_Lease.Room_ID
WHERE (((dbo_Proj_Lease.Room_ID) Is Null));
```

SQL View:

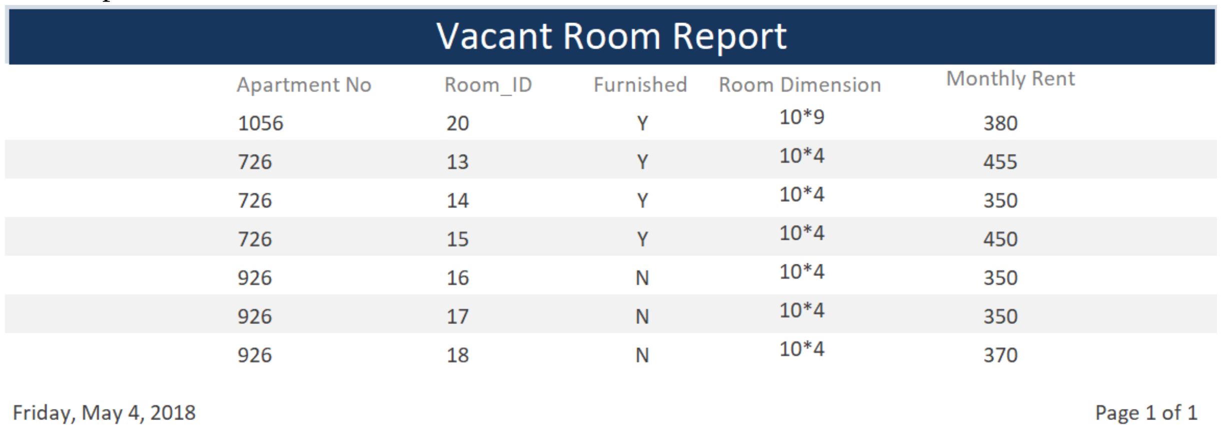
```
SELECT Proj_Room.[Apartment No], Proj_Room.Furnished, Proj_Room.[Monthly Rent],
Proj_Room.[Room Dimension],
Proj_Room.Room_ID
FROM Proj_Room LEFT JOIN Proj_Lease ON Proj_Room.Room_ID = Proj_Lease.Room_ID
WHERE (((Proj_Lease.Room_ID) Is Null));
```



```
SELECT Proj_Room.[Apartment No], Proj_Room.Furnished, Proj_Room.[Monthly Rent], Proj_Room.[Room Dimension],
Proj_Room.Room_ID
FROM Proj_Room LEFT JOIN Proj_Lease ON Proj_Room.Room_ID = Proj_Lease.Room_ID
WHERE (((Proj_Lease.Room_ID) Is Null));
```

	Apartment No	Furnished	Monthly Rent	Room Dimension	Room_ID
1	726	Y	455	10*4	13
2	726	Y	350	10*4	14
3	726	Y	450	10*4	15
4	926	N	350	10*4	16
5	926	N	350	10*4	17
6	926	N	370	10*4	18
7	1056	Y	380	10*9	20

Access Report View:

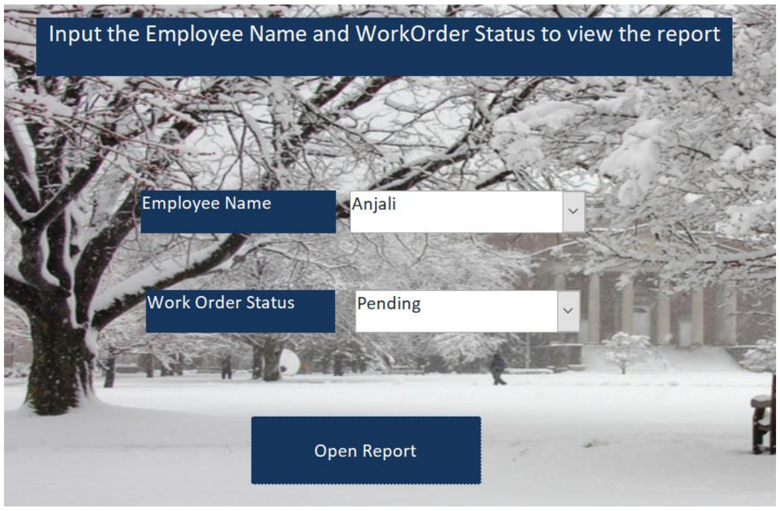


Vacant Room Report

Apartment No	Room_ID	Furnished	Room Dimension	Monthly Rent
1056	20	Y	10*9	380
726	13	Y	10*4	455
726	14	Y	10*4	350
726	15	Y	10*4	450
926	16	N	10*4	350
926	17	N	10*4	350
926	18	N	10*4	370

Friday, May 4, 2018 Page 1 of 1

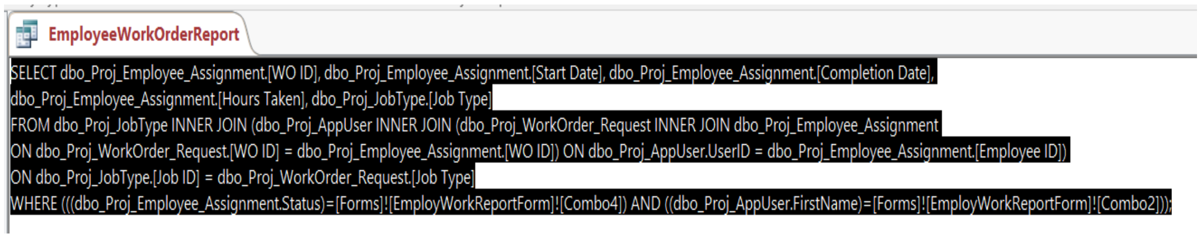
- **Employee Work Order Report:**
  This report will retrieve all the work orders for the selected employee and work order status. This information will help the property owners to analyze the details of the work orders worked upon by the employees. For example: The below report displays the Pending work orders of the employee named Anjali.


Input the Employee Name and WorkOrder Status to view the report

Employee Name: Anjali

Work Order Status: Pending

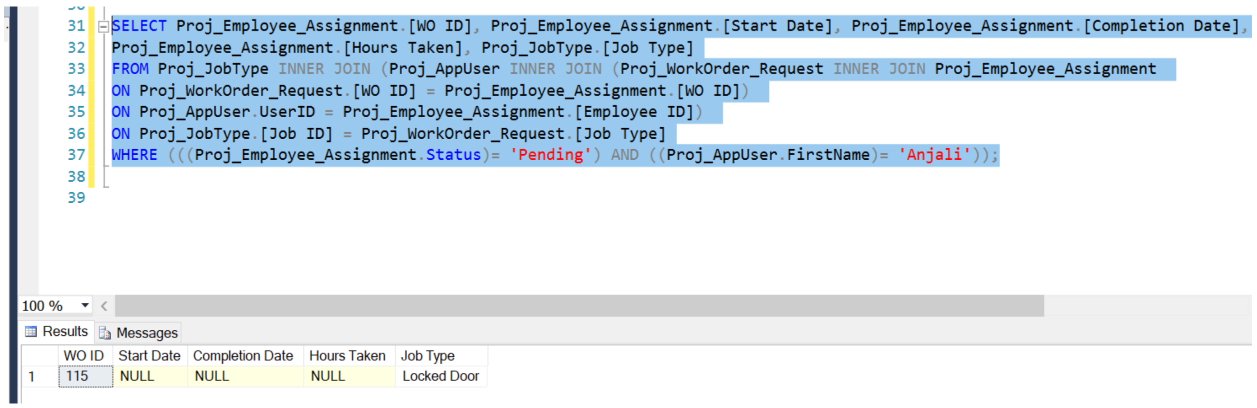
Query Used:
SELECT dbo_Proj_Employee_Assignment.[WO ID], dbo_Proj_Employee_Assignment.[Start Date], dbo_Proj_Employee_Assignment.[Completion Date], dbo_Proj_Employee_Assignment.[Hours Taken], dbo_Proj_JobType.[Job Type]
FROM dbo_Proj_JobType INNER JOIN (dbo_Proj_AppUser INNER JOIN (dbo_Proj_WorkOrder_Request INNER JOIN dbo_Proj_Employee_Assignment
ON dbo_Proj_WorkOrder_Request.[WO ID] = dbo_Proj_Employee_Assignment.[WO ID]) ON dbo_Proj_AppUser.UserID = dbo_Proj_Employee_Assignment.[Employee ID])
ON dbo_Proj_JobType.[Job ID] = dbo_Proj_WorkOrder_Request.[Job Type]
WHERE (((dbo_Proj_Employee_Assignment.Status)=[Forms]![EmployWorkReportForm]![Combo4]) AND ((dbo_Proj_AppUser.FirstName)=[Forms]![EmployWorkReportForm]![Combo2]));



```
SELECT dbo_Proj_Employee_Assignment.[WO ID], dbo_Proj_Employee_Assignment.[Start Date], dbo_Proj_Employee_Assignment.[Completion Date],
dbo_Proj_Employee_Assignment.[Hours Taken], dbo_Proj_JobType.[Job Type]
FROM dbo_Proj_JobType INNER JOIN (dbo_Proj_AppUser INNER JOIN (dbo_Proj_WorkOrder_Request INNER JOIN dbo_Proj_Employee_Assignment
ON dbo_Proj_WorkOrder_Request.[WO ID] = dbo_Proj_Employee_Assignment.[WO ID]) ON dbo_Proj_AppUser.UserID = dbo_Proj_Employee_Assignment.[Employee ID])
ON dbo_Proj_JobType.[Job ID] = dbo_Proj_WorkOrder_Request.[Job Type]
WHERE (((dbo_Proj_Employee_Assignment.Status)=[Forms]![EmployWorkReportForm]![Combo4]) AND ((dbo_Proj_AppUser.FirstName)=[Forms]![EmployWorkReportForm]![Combo2]));
```

SQL View:

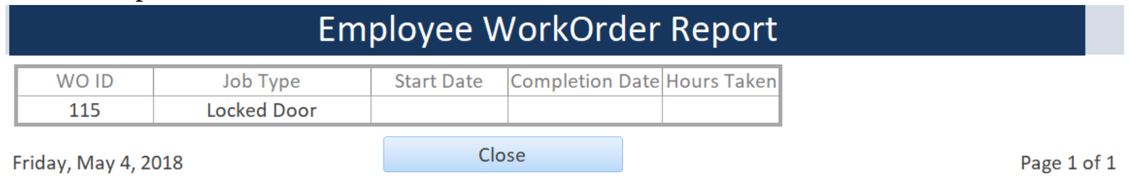
```
SELECT Proj_Employee_Assignment.[WO ID], Proj_Employee_Assignment.[Start Date],
Proj_Employee_Assignment.[Completion Date],
Proj_Employee_Assignment.[Hours Taken], Proj_JobType.[Job Type]
FROM Proj_JobType INNER JOIN (Proj_AppUser INNER JOIN (Proj_WorkOrder_Request
INNER JOIN Proj_Employee_Assignment
ON Proj_WorkOrder_Request.[WO ID] = Proj_Employee_Assignment.[WO ID])
ON Proj_AppUser.UserID = Proj_Employee_Assignment.[Employee ID])
ON Proj_JobType.[Job ID] = Proj_WorkOrder_Request.[Job Type]
WHERE (((Proj_Employee_Assignment.Status)= 'Pending') AND
((Proj_AppUser.FirstName)= 'Anjali'));
```



```
SELECT Proj_Employee_Assignment.[WO ID], Proj_Employee_Assignment.[Start Date], Proj_Employee_Assignment.[Completion Date],
Proj_Employee_Assignment.[Hours Taken], Proj_JobType.[Job Type]
FROM Proj_JobType INNER JOIN (Proj_AppUser INNER JOIN (Proj_WorkOrder_Request INNER JOIN Proj_Employee_Assignment
ON Proj_WorkOrder_Request.[WO ID] = Proj_Employee_Assignment.[WO ID])
ON Proj_AppUser.UserID = Proj_Employee_Assignment.[Employee ID])
ON Proj_JobType.[Job ID] = Proj_WorkOrder_Request.[Job Type]
WHERE (((Proj_Employee_Assignment.Status)= 'Pending') AND ((Proj_AppUser.FirstName)= 'Anjali'));
```

	WO ID	Start Date	Completion Date	Hours Taken	Job Type
1	115	NULL	NULL	NULL	Locked Door

Access Report View:

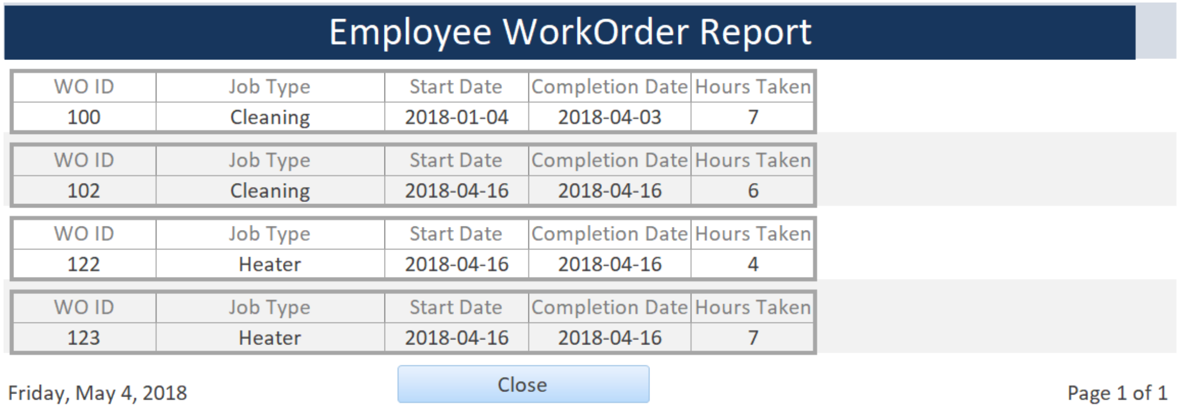


Employee WorkOrder Report

WO ID	Job Type	Start Date	Completion Date	Hours Taken
115	Locked Door			

Friday, May 4, 2018 Page 1 of 1

(By selecting Employee Name: Anjali and status Pending)



Employee WorkOrder Report

WO ID	Job Type	Start Date	Completion Date	Hours Taken
100	Cleaning	2018-01-04	2018-04-03	7
102	Cleaning	2018-04-16	2018-04-16	6
122	Heater	2018-04-16	2018-04-16	4
123	Heater	2018-04-16	2018-04-16	7

Friday, May 4, 2018 Page 1 of 1

(By selecting Employee Name: Aditya and status Complete)

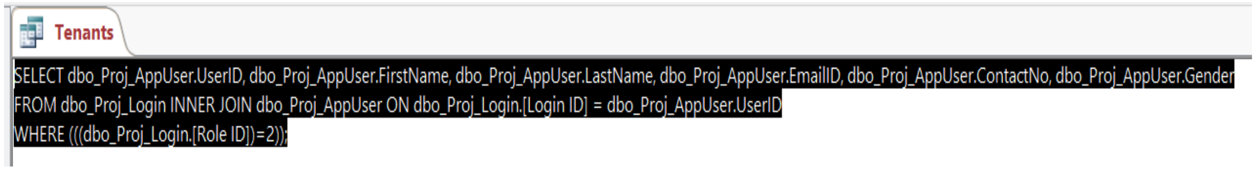
ii)Tenant queries the database for:

Retrieving the Profile details of the Tenants:

This data question allows tenants to get all the details about their respective contact details which can be edited by the tenant whenever required. The tenants can also view their lease and work order details in the same view. This will help them in retrieving a comprehensive view of tenant's information. Based upon the tenants retrieved in the below query, the respective lease and work order details would be retrieved in the sub form.

Query Used:

SELECT dbo_Proj_AppUser.UserID, dbo_Proj_AppUser.FirstName, dbo_Proj_AppUser.LastName, dbo_Proj_AppUser.EmailID, dbo_Proj_AppUser.ContactNo, dbo_Proj_AppUser.Gender FROM dbo_Proj_Login INNER JOIN dbo_Proj_AppUser ON dbo_Proj_Login.[Login ID] = dbo_Proj_AppUser.UserID WHERE (((dbo_Proj_Login.[Role ID])=2))

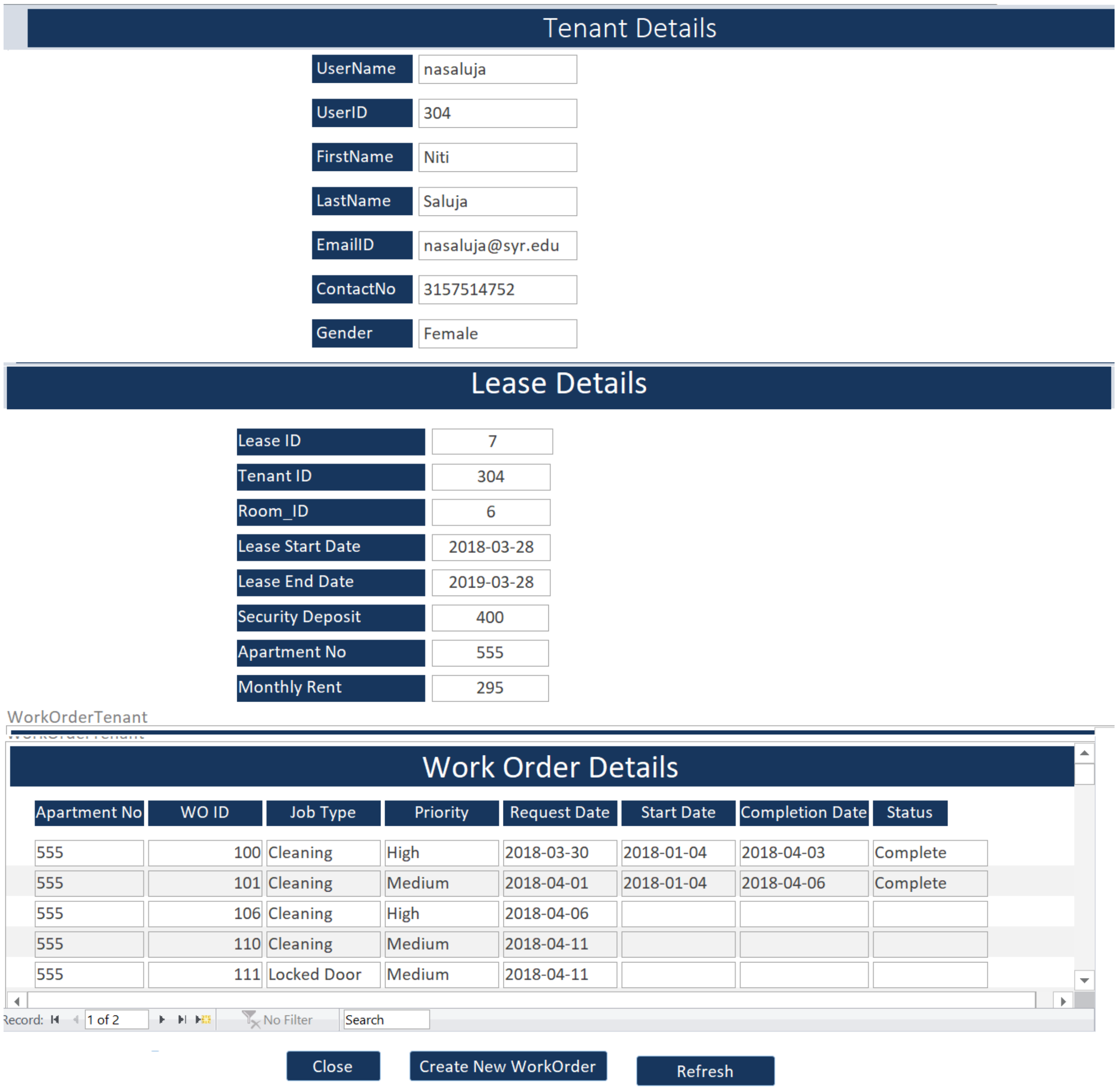


```
SELECT dbo_Proj_AppUser.UserID, dbo_Proj_AppUser.FirstName, dbo_Proj_AppUser.LastName, dbo_Proj_AppUser.EmailID, dbo_Proj_AppUser.ContactNo, dbo_Proj_AppUser.Gender
FROM dbo_Proj_Login INNER JOIN dbo_Proj_AppUser ON dbo_Proj_Login.[Login ID] = dbo_Proj_AppUser.UserID
WHERE (((dbo_Proj_Login.[Role ID])=2));
```

Select * from dbo_Proj_Lease;

Select * from dbo_Proj_WorkOrder;

Access Report View:



Tenant Details

UserName	nasaluja
UserID	304
FirstName	Niti
LastName	Saluja
EmailID	nasaluja@syr.edu
ContactNo	3157514752
Gender	Female

Lease Details

Lease ID	7
Tenant ID	304
Room_ID	6
Lease Start Date	2018-03-28
Lease End Date	2019-03-28
Security Deposit	400
Apartment No	555
Monthly Rent	295

WorkOrderTenant

Work Order Details

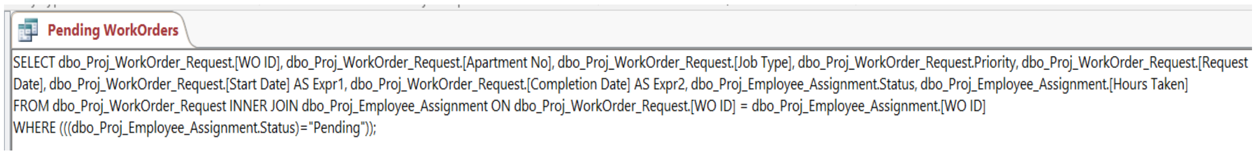
Apartment No	WO ID	Job Type	Priority	Request Date	Start Date	Completion Date	Status
555	100	Cleaning	High	2018-03-30	2018-01-04	2018-04-03	Complete
555	101	Cleaning	Medium	2018-04-01	2018-01-04	2018-04-06	Complete
555	106	Cleaning	High	2018-04-06			
555	110	Cleaning	Medium	2018-04-11			
555	111	Locked Door	Medium	2018-04-11			

<u>iii)Employee queries the database for:</u>

Pending Work Orders:
This will allow the employees to retrieve all details about the pending work orders, which they can be updated on completion.

Query Used:

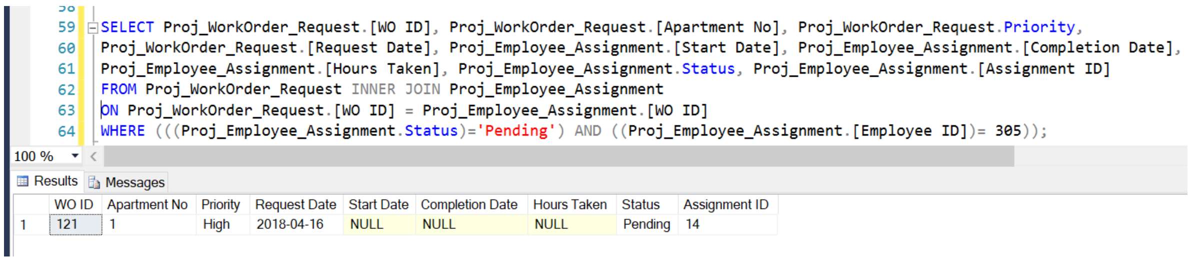
SELECT dbo_Proj_WorkOrder_Request.[WO ID], dbo_Proj_WorkOrder_Request.[Apartment No], dbo_Proj_WorkOrder_Request.[Job Type], dbo_Proj_WorkOrder_Request.Priority, dbo_Proj_WorkOrder_Request.[Request Date], dbo_Proj_WorkOrder_Request.[Start Date] AS Expr1, dbo_Proj_WorkOrder_Request.[Completion Date] AS Expr2, dbo_Proj_Employee_Assignment.Status, dbo_Proj_Employee_Assignment.[Hours Taken]
FROM dbo_Proj_WorkOrder_Request INNER JOIN dbo_Proj_Employee_Assignment ON dbo_Proj_WorkOrder_Request.[WO ID] = dbo_Proj_Employee_Assignment.[WO ID] WHERE (((dbo_Proj_Employee_Assignment.Status)="Pending"));



```
SELECT dbo_Proj_WorkOrder_Request.[WO ID], dbo_Proj_WorkOrder_Request.[Apartment No], dbo_Proj_WorkOrder_Request.[Job Type], dbo_Proj_WorkOrder_Request.Priority, dbo_Proj_WorkOrder_Request.[Request Date], dbo_Proj_WorkOrder_Request.[Start Date] AS Expr1, dbo_Proj_WorkOrder_Request.[Completion Date] AS Expr2, dbo_Proj_Employee_Assignment.Status, dbo_Proj_Employee_Assignment.[Hours Taken]
FROM dbo_Proj_WorkOrder_Request INNER JOIN dbo_Proj_Employee_Assignment ON dbo_Proj_WorkOrder_Request.[WO ID] = dbo_Proj_Employee_Assignment.[WO ID]
WHERE (((dbo_Proj_Employee_Assignment.Status)="Pending"));
```

SQL View:

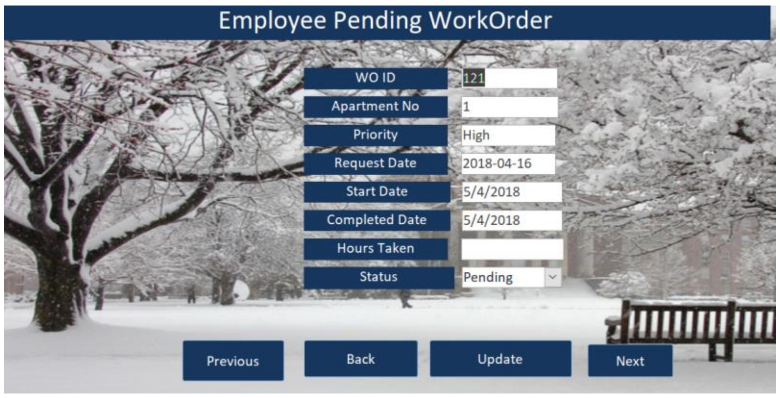
```
SELECT Proj_WorkOrder_Request.[WO ID], Proj_WorkOrder_Request.[Apartment No],
Proj_WorkOrder_Request.Priority,
Proj_WorkOrder_Request.[Request Date], Proj_Employee_Assignment.[Start Date],
Proj_Employee_Assignment.[Completion Date],
Proj_Employee_Assignment.[Hours Taken], Proj_Employee_Assignment.Status,
Proj_Employee_Assignment.[Assignment ID]
FROM Proj_WorkOrder_Request INNER JOIN Proj_Employee_Assignment
ON Proj_WorkOrder_Request.[WO ID] = Proj_Employee_Assignment.[WO ID]
WHERE (((Proj_Employee_Assignment.Status)='Pending') AND
((Proj_Employee_Assignment.[Employee ID])= 305));
```



```
SELECT Proj_WorkOrder_Request.[WO ID], Proj_WorkOrder_Request.[Apartment No], Proj_WorkOrder_Request.Priority,
Proj_WorkOrder_Request.[Request Date], Proj_Employee_Assignment.[Start Date], Proj_Employee_Assignment.[Completion Date],
Proj_Employee_Assignment.[Hours Taken], Proj_Employee_Assignment.Status, Proj_Employee_Assignment.[Assignment ID]
FROM Proj_WorkOrder_Request INNER JOIN Proj_Employee_Assignment
ON Proj_WorkOrder_Request.[WO ID] = Proj_Employee_Assignment.[WO ID]
WHERE (((Proj_Employee_Assignment.Status)='Pending') AND ((Proj_Employee_Assignment.[Employee ID])= 305));
```

	WO ID	Apartment No	Priority	Request Date	Start Date	Completion Date	Hours Taken	Status	Assignment ID
1	121	1	High	2018-04-16	NULL	NULL	NULL	Pending	14

Access Form View:

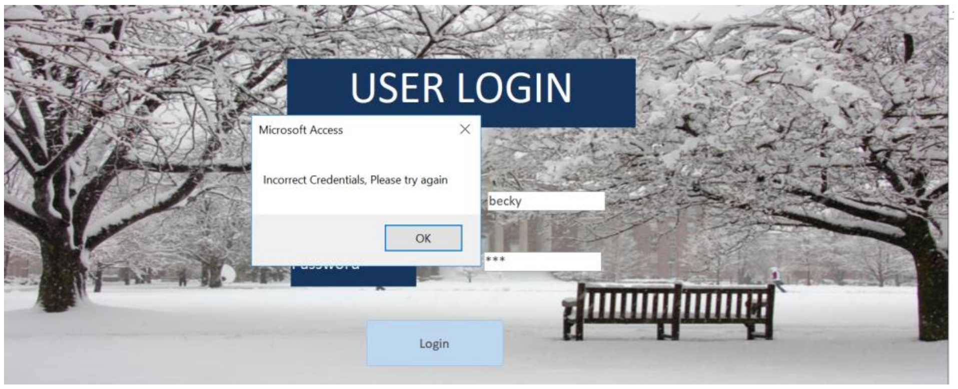


WO ID	121
Apartment No	1
Priority	High
Request Date	2018-04-16
Start Date	5/4/2018
Completed Date	5/4/2018
Hours Taken	
Status	Pending

INTERFACES:

1. Landlord Flow:

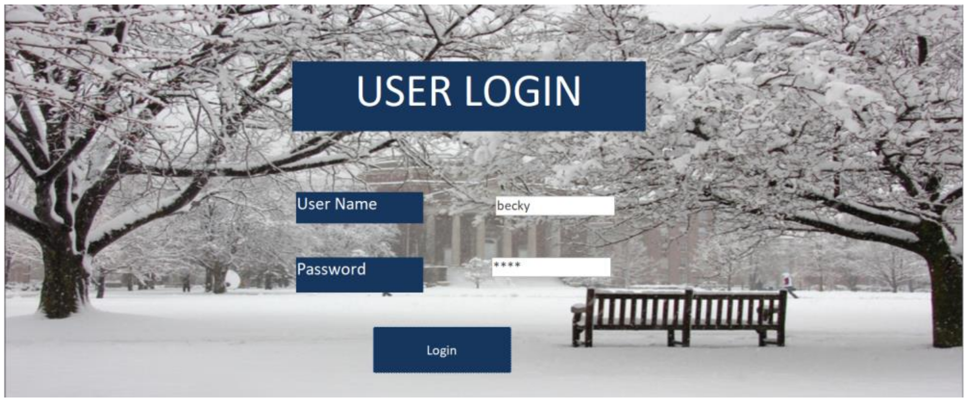
Login Form:



Incorrect Credentials, Please try again

becky

***

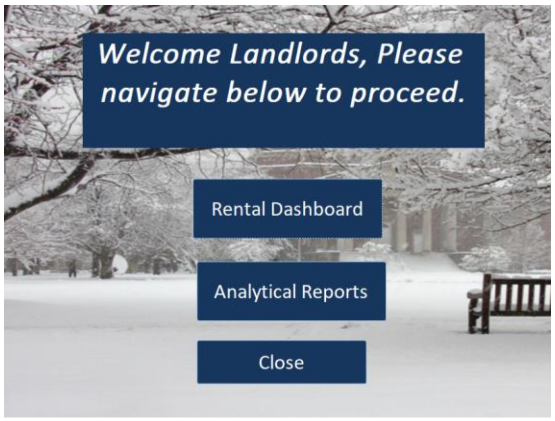


becky

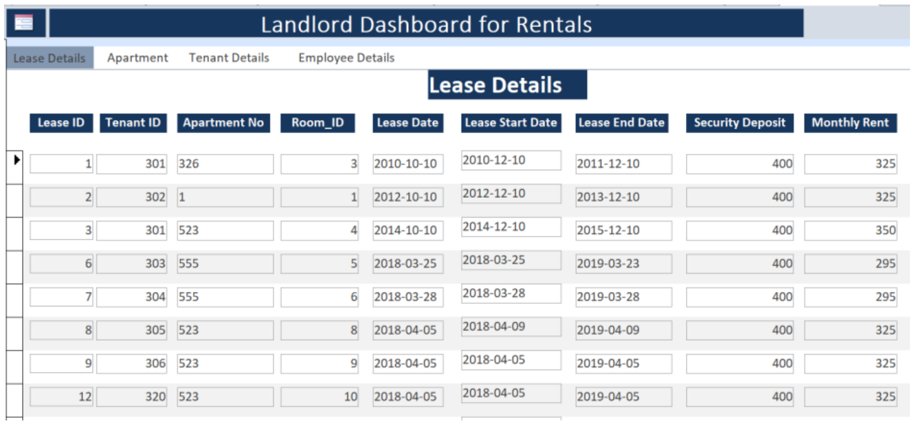
****

(Correct Credentials)

Landlord Dashboard



Lease Details View for Landlords:

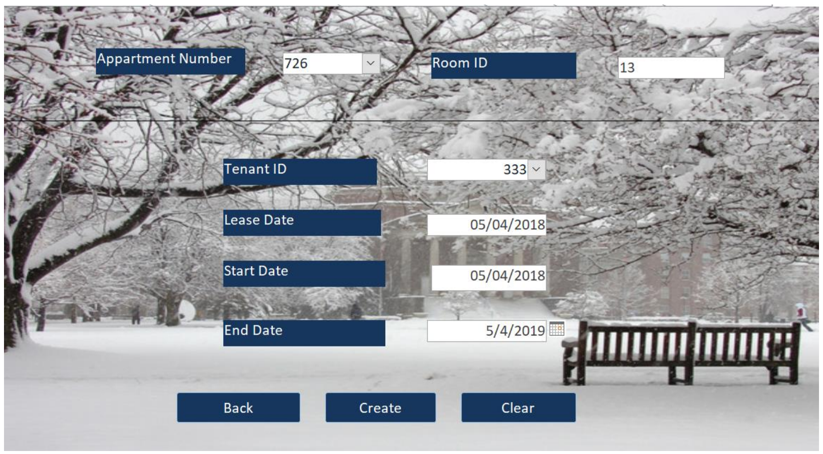


Lease Details

Lease ID	Tenant ID	Apartment No	Room_ID	Lease Date	Lease Start Date	Lease End Date	Security Deposit	Monthly Rent
1	301	326	3	2010-10-10	2010-12-10	2011-12-10	400	325
2	302	1	1	2012-10-10	2012-12-10	2013-12-10	400	325
3	301	523	4	2014-10-10	2014-12-10	2015-12-10	400	350
6	303	555	5	2018-03-25	2018-03-25	2019-03-23	400	295
7	304	555	6	2018-03-28	2018-03-28	2019-03-28	400	295
8	305	523	8	2018-04-05	2018-04-09	2019-04-09	400	325
9	306	523	9	2018-04-05	2018-04-05	2019-04-05	400	325
12	320	523	10	2018-04-05	2018-04-05	2019-04-05	400	325

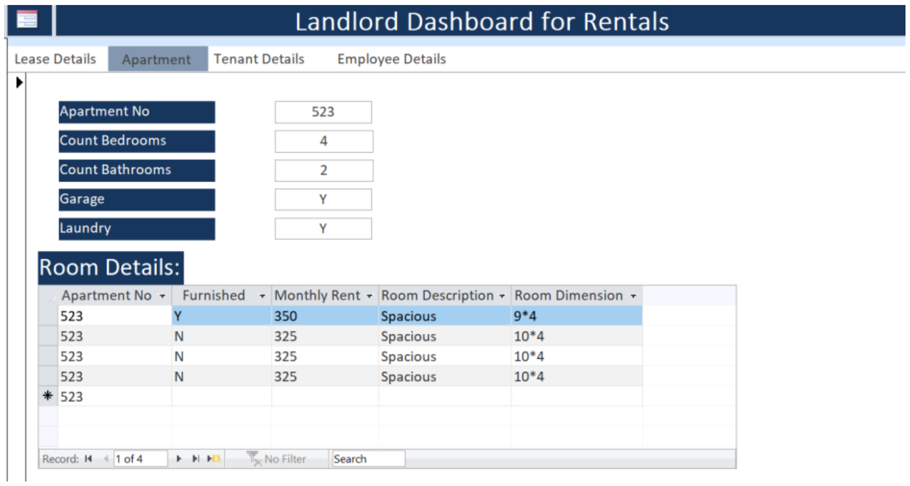
Creating new Lease:

The combo box for Apartment Number will display the available apartments with the respective room numbers so that the lease is created only for the vacant rooms.



Appartment Number	726
Room ID	13
Tenant ID	333
Lease Date	05/04/2018
Start Date	05/04/2018
End Date	5/4/2019

Apartment Details for Landlords:

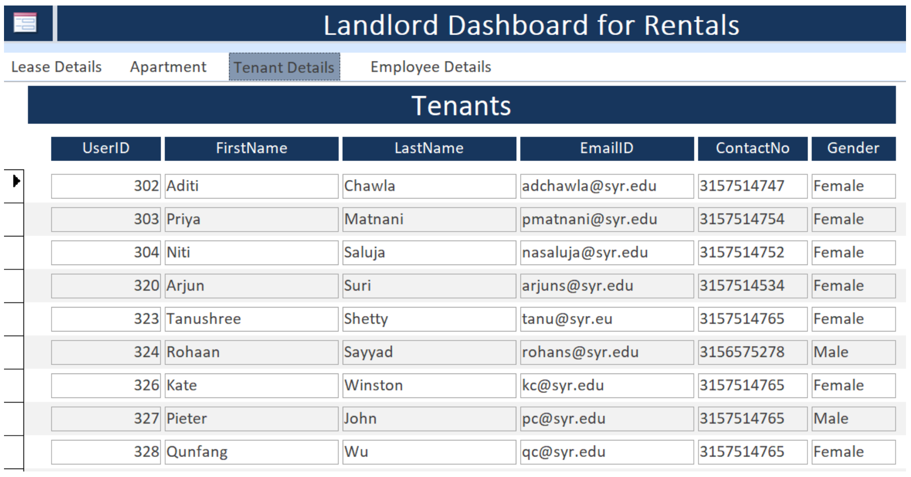


Apartment No	523
Count Bedrooms	4
Count Bathrooms	2
Garage	Y
Laundry	Y

Room Details:

Apartment No	Furnished	Monthly Rent	Room Description	Room Dimension
523	Y	350	Spacious	9*4
523	N	325	Spacious	10*4
523	N	325	Spacious	10*4
523	N	325	Spacious	10*4
523				

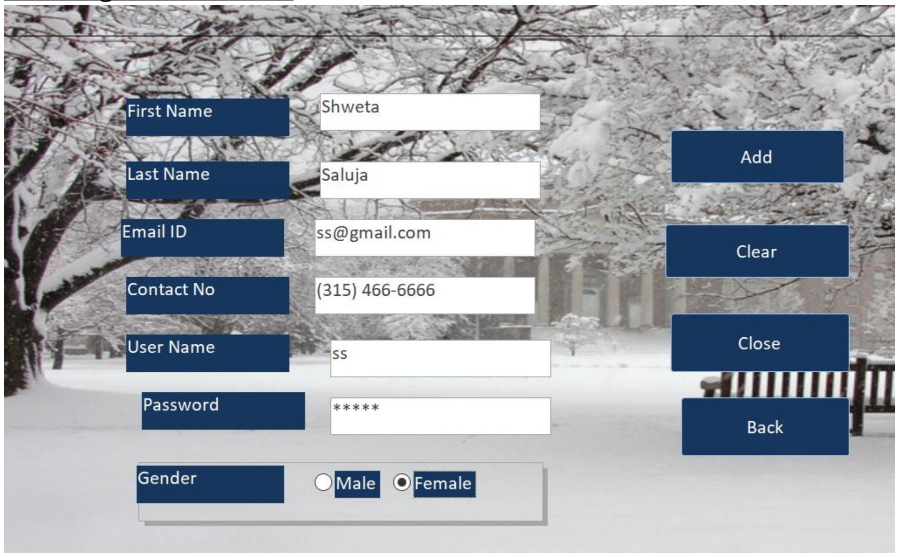
Tenant Details View for Landlords:



Tenants

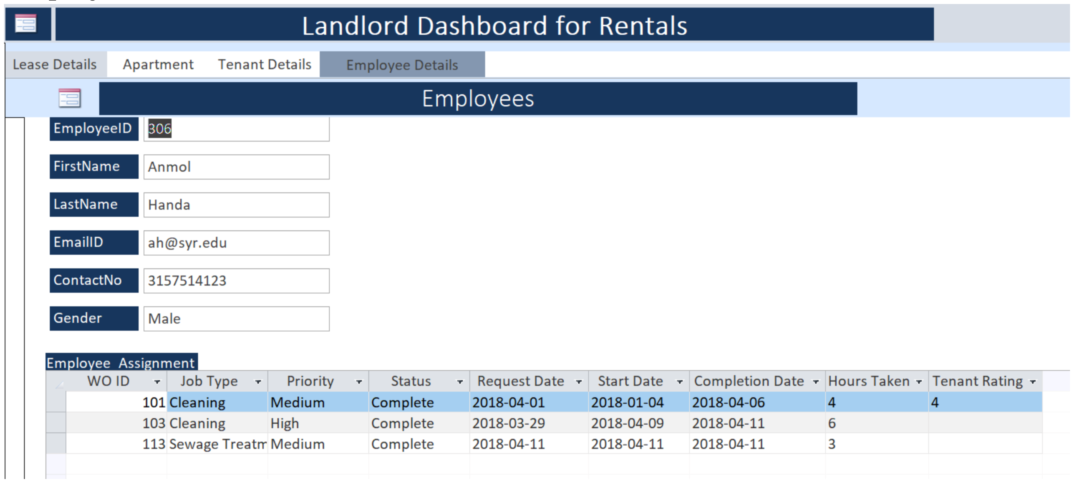
UserID	FirstName	LastName	EmailID	ContactNo	Gender
302	Aditi	Chawla	adchawla@syr.edu	3157514747	Female
303	Priya	Matnani	pmatnani@syr.edu	3157514754	Female
304	Niti	Saluja	nasaluja@syr.edu	3157514752	Female
320	Arjun	Suri	arjuns@syr.edu	3157514534	Female
323	Tanushree	Shetty	tanu@syr.eu	3157514765	Female
324	Rohaan	Sayyad	rohans@syr.edu	3156575278	Male
326	Kate	Winston	kc@syr.edu	3157514765	Female
327	Pieter	John	pc@syr.edu	3157514765	Male
328	Qunfang	Wu	qc@syr.edu	3157514765	Female

Creating a new tenant:



First Name	Shweta
Last Name	Saluja
Email ID	ss@gmail.com
Contact No	(315) 466-6666
User Name	ss
Password	*****
Gender	Male [✓ Female]

Employee Details View for Landlord:



Employees

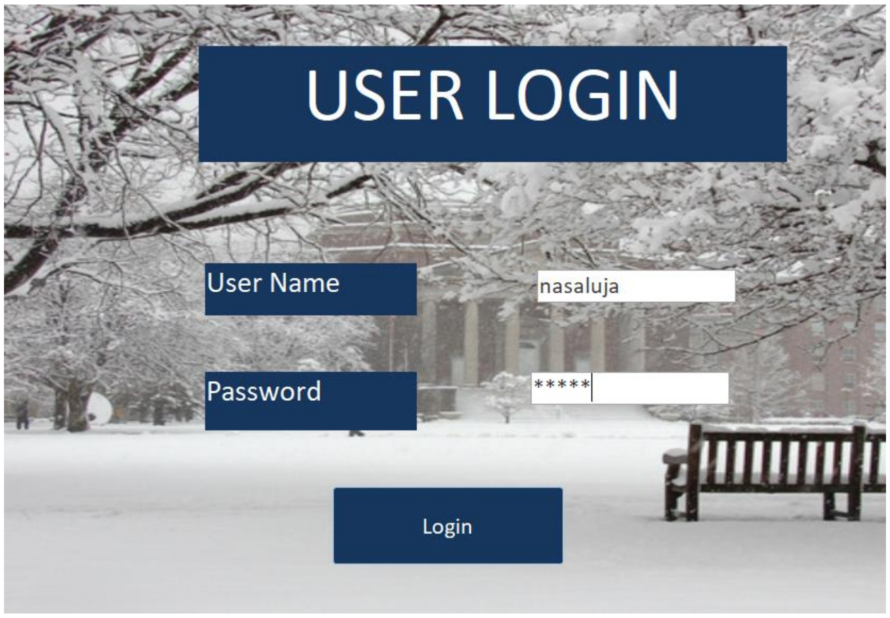
EmployeeID	306
FirstName	Anmol
LastName	Handa
EmailID	ah@syr.edu
ContactNo	3157514123
Gender	Male

Employee_Assignment

WO ID	Job Type	Priority	Status	Request Date	Start Date	Completion Date	Hours Taken	Tenant Rating
101	Cleaning	Medium	Complete	2018-04-01	2018-01-04	2018-04-06	4	4
103	Cleaning	High	Complete	2018-03-29	2018-04-09	2018-04-11	6	
113	Sewage Treatm	Medium	Complete	2018-04-11	2018-04-11	2018-04-11	3	

2. Tenant Flow:

Login Page:

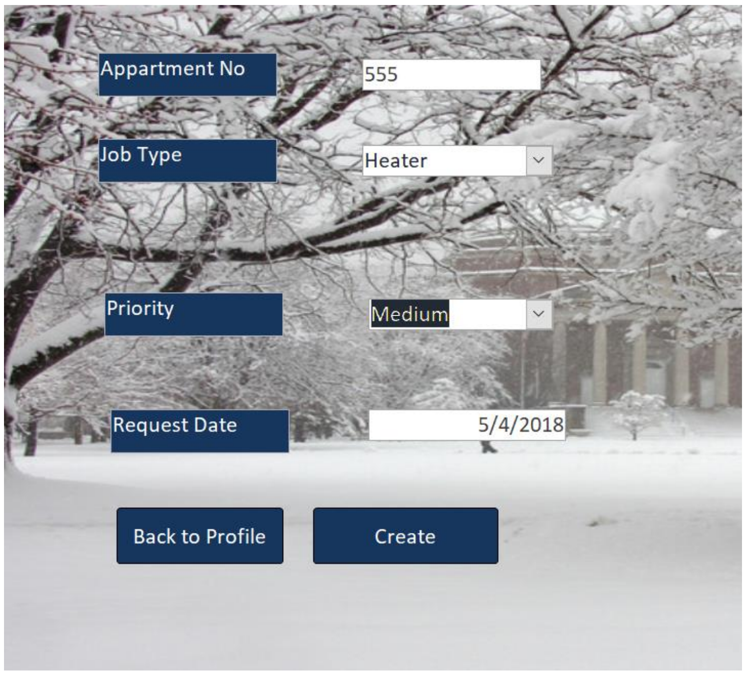


USER LOGIN

User Name: nasaluja

Password: *****

Creating new Work Order:



Appartment No: 555

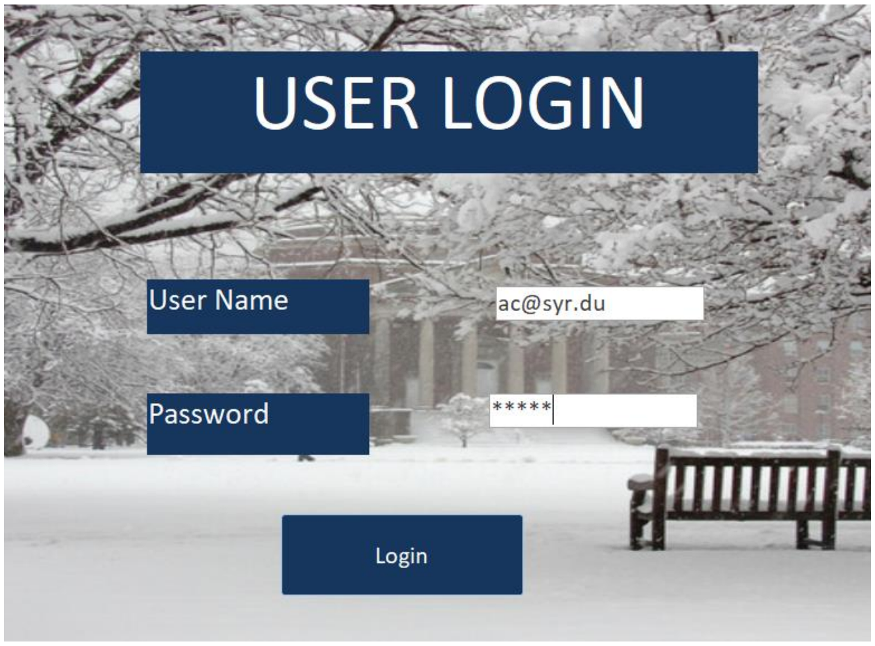
Job Type: Heater

Priority: Medium

Request Date: 5/4/2018

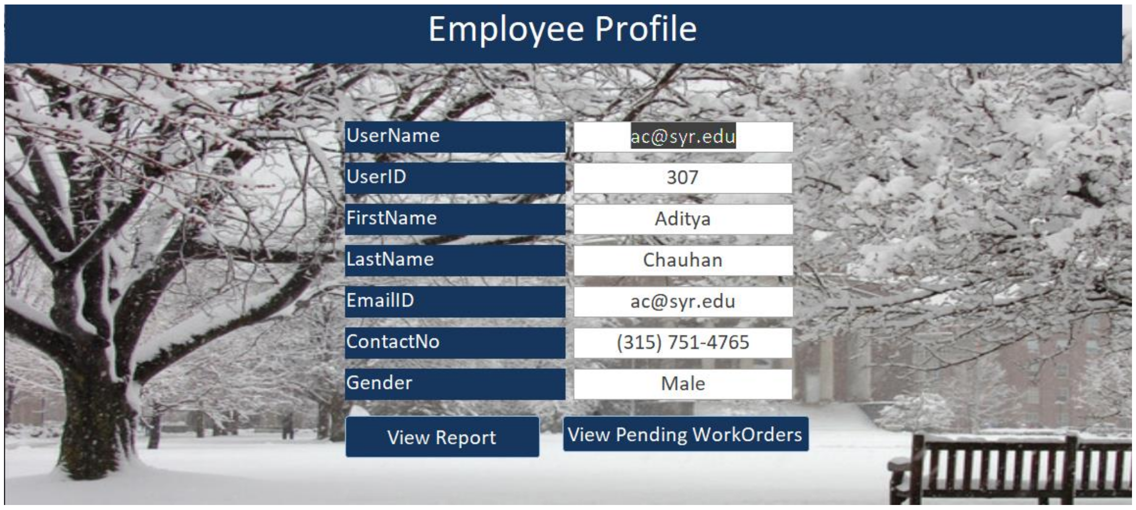
3. Employee Flow:

Login Page:



User Name	ac@syr.du
Password	*****

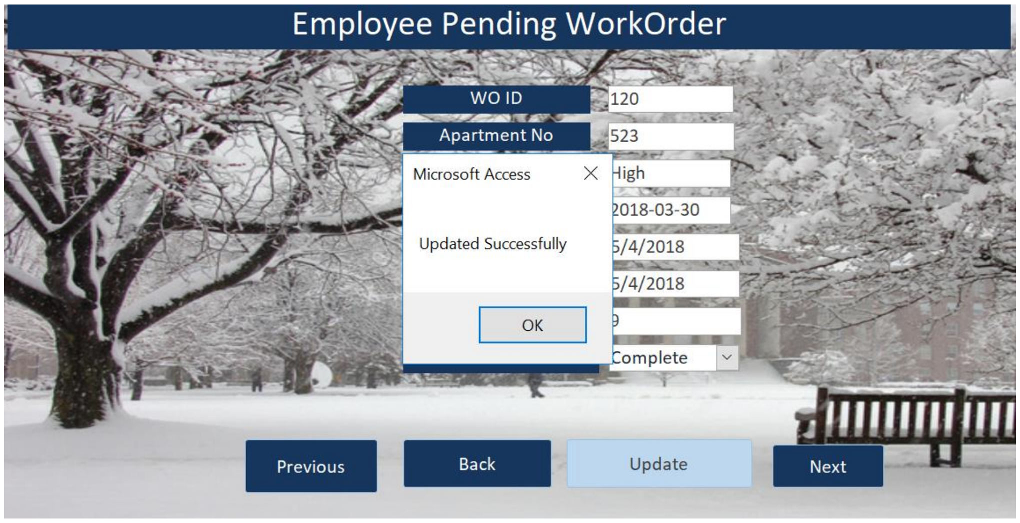
Employee Profile:



Employee Profile

UserName	ac@syr.edu
UserID	307
FirstName	Aditya
LastName	Chauhan
EmailID	ac@syr.edu
ContactNo	(315) 751-4765
Gender	Male

Updating Pending Work Orders:

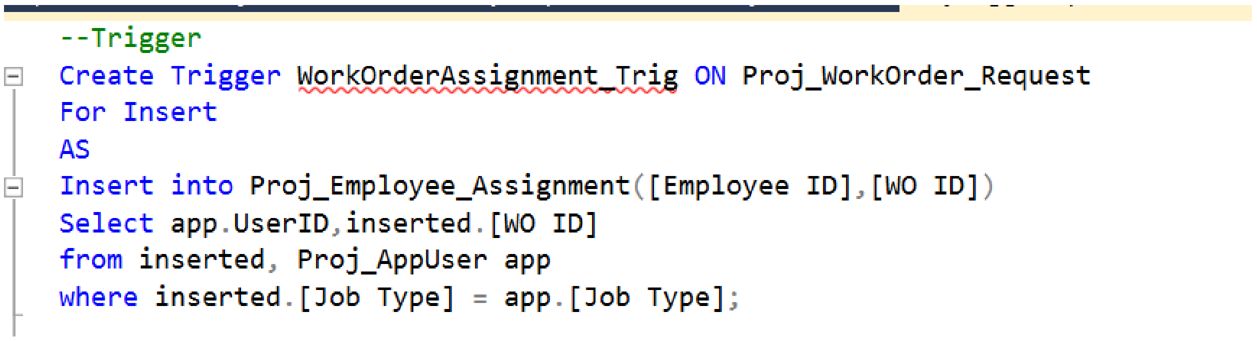


WO ID 120
Apartment No 523
High
2018-03-30
5/4/2018
5/4/2018
9
Complete

Microsoft Access
Updated Successfully

TRIGGER:

The trigger auto assigns a newly created work order to an employee (i.e. It inserts a new record in the Employee Assignment table each time a new record is created in the Work Order table)

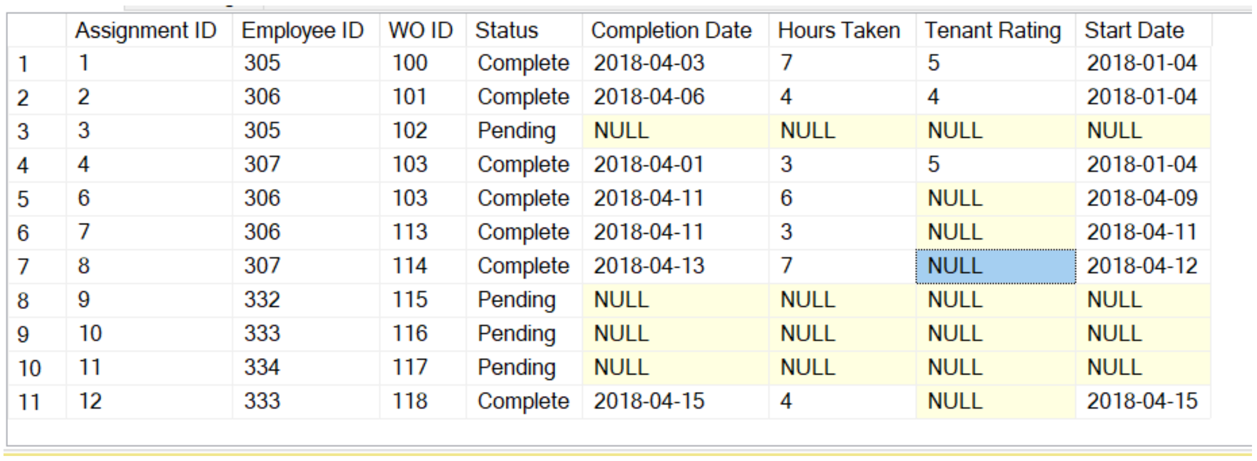


```
--Trigger
Create Trigger WorkOrderAssignment_Trig ON Proj_WorkOrder_Request
For Insert
AS
Insert into Proj_Employee_Assignment([Employee ID],[WO ID])
Select app.UserID,inserted.[WO ID]
from inserted, Proj_AppUser app
where inserted.[Job Type] = app.[Job Type];
```

```
--Trigger
  Create Trigger WorkOrderAssignment_Trig ON Proj_WorkOrder_Request
  For Insert
  AS
  Insert into Proj_Employee_Assignment([Employee ID],[WO ID])
  Select app.UserID,inserted.[WO ID]
  from inserted, Proj_AppUser app
  where inserted.[Job Type] = app.[Job Type];


  Select * from Proj_WorkOrder_Request
  Select * from Proj_Employee_Assignment
```

Employee_Assignment table before the trigger

```
Select * from Proj_Employee_Assignment
```



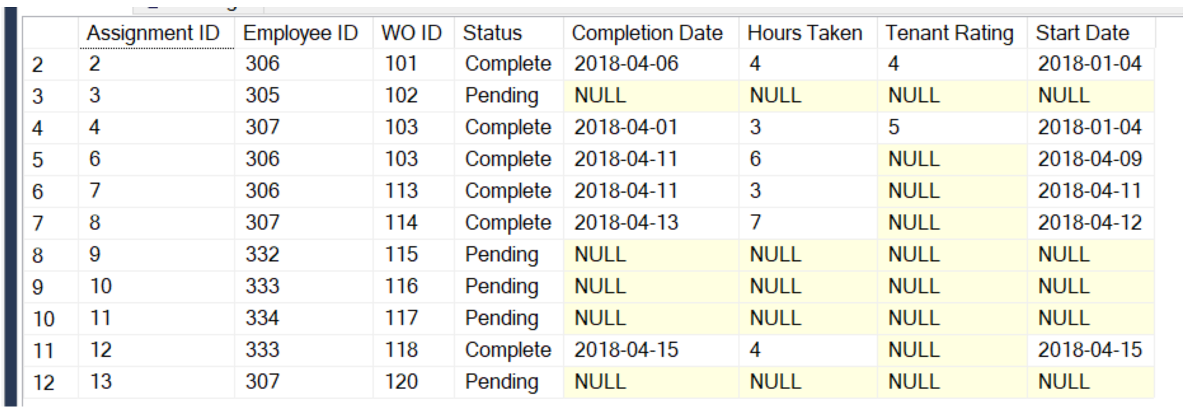
	Assignment ID	Employee ID	WO ID	Status	Completion Date	Hours Taken	Tenant Rating	Start Date
1	1	305	100	Complete	2018-04-03	7	5	2018-01-04
2	2	306	101	Complete	2018-04-06	4	4	2018-01-04
3	3	305	102	Pending	NULL	NULL	NULL	NULL
4	4	307	103	Complete	2018-04-01	3	5	2018-01-04
5	6	306	103	Complete	2018-04-11	6	NULL	2018-04-09
6	7	306	113	Complete	2018-04-11	3	NULL	2018-04-11
7	8	307	114	Complete	2018-04-13	7	NULL	2018-04-12
8	9	332	115	Pending	NULL	NULL	NULL	NULL
9	10	333	116	Pending	NULL	NULL	NULL	NULL
10	11	334	117	Pending	NULL	NULL	NULL	NULL
11	12	333	118	Complete	2018-04-15	4	NULL	2018-04-15

Employee_Assignment table after the trigger

The WorkOrder is auto assigned to Employee 307 based on his Job Type and a Pending WorkOrder is created in the Assignment table.

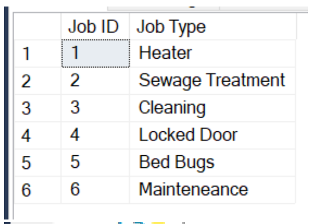
Insert into [Proj_WorkOrder_Request]([Apartment No],[Job Type],Priority, [Request Date]) values (523,3,'High','03/30/2018')

```
Select * from Proj_Employee_Assignment
```

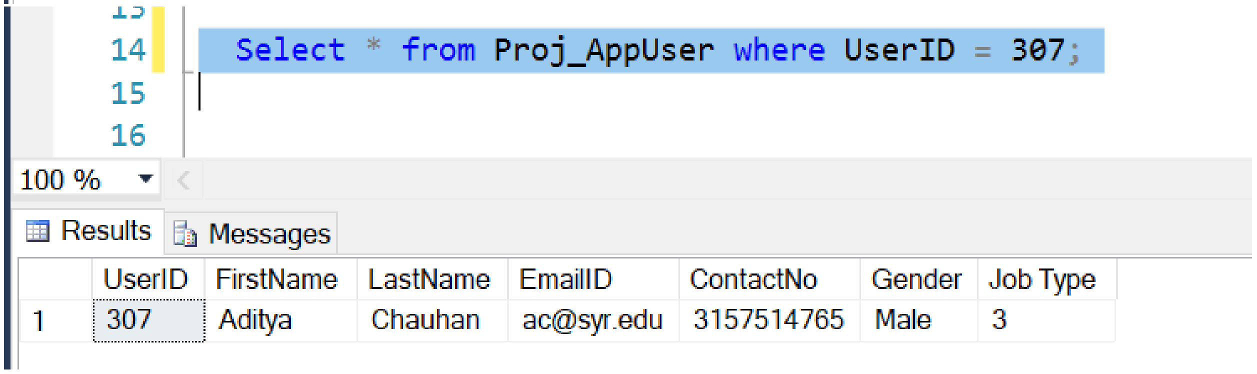


	Assignment ID	Employee ID	WO ID	Status	Completion Date	Hours Taken	Tenant Rating	Start Date
2	2	306	101	Complete	2018-04-06	4	4	2018-01-04
3	3	305	102	Pending	NULL	NULL	NULL	NULL
4	4	307	103	Complete	2018-04-01	3	5	2018-01-04
5	6	306	103	Complete	2018-04-11	6	NULL	2018-04-09
6	7	306	113	Complete	2018-04-11	3	NULL	2018-04-11
7	8	307	114	Complete	2018-04-13	7	NULL	2018-04-12
8	9	332	115	Pending	NULL	NULL	NULL	NULL
9	10	333	116	Pending	NULL	NULL	NULL	NULL
10	11	334	117	Pending	NULL	NULL	NULL	NULL
11	12	333	118	Complete	2018-04-15	4	NULL	2018-04-15
12	13	307	120	Pending	NULL	NULL	NULL	NULL

```
Select * from Proj_JobType
```



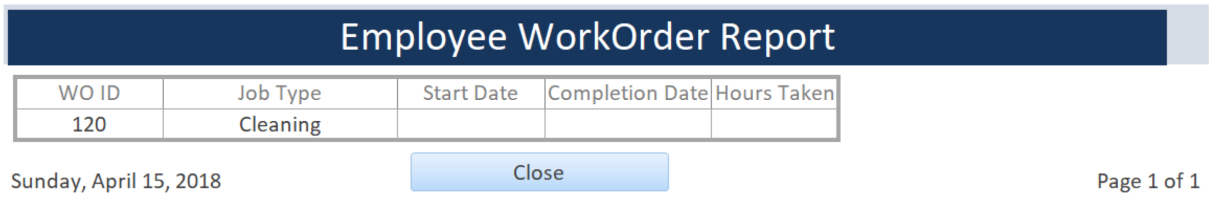
	Job ID	Job Type
1	1	Heater
2	2	Sewage Treatment
3	3	Cleaning
4	4	Locked Door
5	5	Bed Bugs
6	6	Mainteneance



```
Select * from Proj_AppUser where UserID = 307;
```

	UserID	FirstName	LastName	EmailID	ContactNo	Gender	Job Type
1	307	Aditya	Chauhan	ac@syr.edu	3157514765	Male	3

Access View:



Employee WorkOrder Report

WO ID	Job Type	Start Date	Completion Date	Hours Taken
120	Cleaning			

Sunday, April 15, 2018

Page 1 of 1

Since the employee 307 is specialized in job type 3 (Cleaning), the newly created work order of job type 3 is auto assigned to 307.