Database & Navigation Properties Lab (15 pts)
Web-Enabled Database Programming Name __Mike Hedges_____

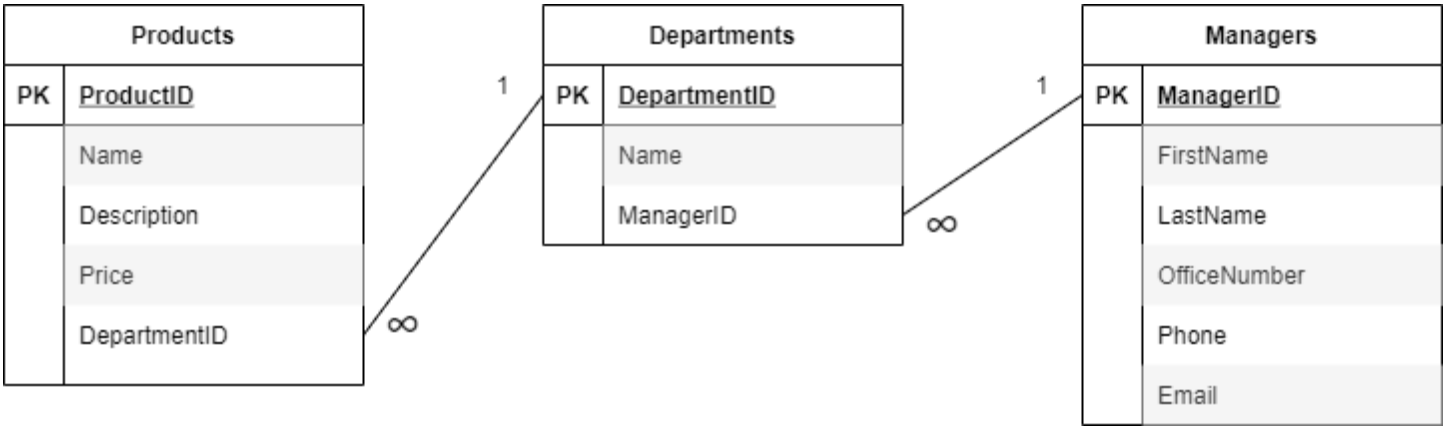
Problem 1:

Let's say you plan to create an application that lists information about products available at Joe's Department Store. Each product has a name, description, price, and department (hardware, office supplies, clothing, shoes, grocery) so your database will eventually include a table called Products. This table should include an integer called ProductID for each product and this column will serve as the primary key.

Instead of saving the string that identifies the department in the same table as all of the other product information, you want a separate Departments table to hold the actual names of the departments. This table should also have a primary key, called DepartmentID in this case.

This application should also keep track of the managers of each department so there should be a table called Managers. It should have a primary key called ManagerID. Each department has only one manager but it's possible for a manager to manage more than one department at a time.

Create a database diagram the includes all three tables and indicates the type of relationships between them (one-to-one, one-to-many).



Problem 1 Continued:

Fill in the code below to create some model classes that can be used with this application. All of your properties should be **auto-implemented**.
Be sure and use the **singular form** of the entities for the names of the classes. And don't forget to include the **navigation** properties!

```
public class Product
{
        public int ProductID {get; set;}
        public string Name {get; set;}
        public string Description {get; set;}
        public float Price {get; set;}
        public virtual Department Department {get; set;}
}
public class Department
{
        public int DepartmentID {get; set;}
        public string Name {get; set;}
        public virtual ICollection<Product>Products {get; set;}
        public virtual Manager Manager {get; set;}
}
public class Manager
{
        public int ManagerID {get; set;}
        public string FirstName {get; set;}
        public string LastName {get; set;}
        public int OfficeNumber {get; set;}
        public int Phone {get; set;}
        public string Email {get; set;}
        public virtual ICollection<Department>Departments {get; set;}
}
```

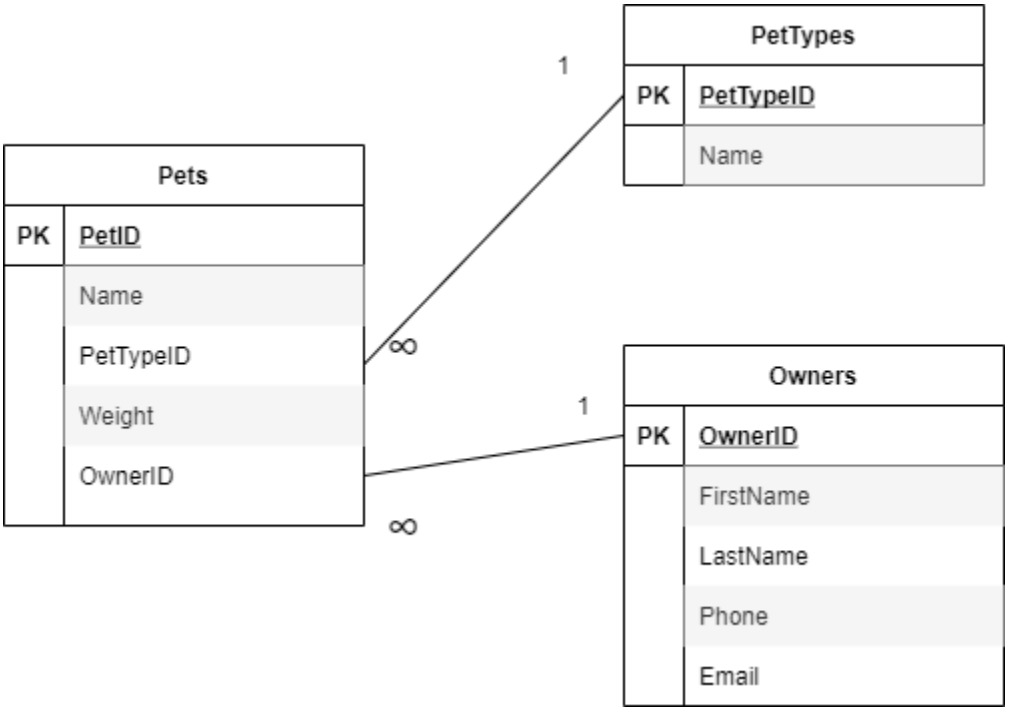
Problem 2:

Let's say you plan to create an application that lists information about pets staying at the Creature Comforts Pet Hotel. Each pet has a name, pet type (dog, cat, etc.), weight, and owner so your database will eventually include a table called Pets. This table should include an integer called PetID for each pet and this column will serve as the primary key.

Instead of saving the string that identifies the pet type in the same table as all of the other pet information, you want a separate PetTypes table to hold the actual types of pets (dog, cat, etc.). This table should also have a primary key, called PetTypeID.

This application should also keep track of the owners of each pet so there should be a table called Owners. It should have a primary key called OwnerID. Each owner can have more than one pet but each pet can have only one owner. Other Owner properties should include FirstName, LastName, Phone, and Email.

Create a database diagram the includes all three tables and indicates the type of relationships between them (one-to-one, one-to-many).



Problem 2 Continued:

Fill in the code below to create some model classes that can be used with this application. All of your properties should be **auto-implemented**.
Be sure and use the **singular form** of the entities for the names of the classes. And don't forget to include the **navigation** properties!

```
public class Pet
{
        public int PetID {get; set;}
        public string Name {get; set;}
        public float Weight {get; set;}
        public virtual PetType PetType {get; set;}
        public virtual Owner Owner {get; set;}
}
public class PetType
{
        public int PetTypeID {get; set;}
        public string Name {get; set;}
        public ICollection<Pet> Pets {get; set;}
}
public class Owner
{
        public int OwnerID {get; set;}
        public string FirstName {get; set;}
        public string LastName {get; set;}
        public int Phone {get; set;}
        public string Email {get; set;}
        public ICollection<Pet> Pets {get; set;}
}
```