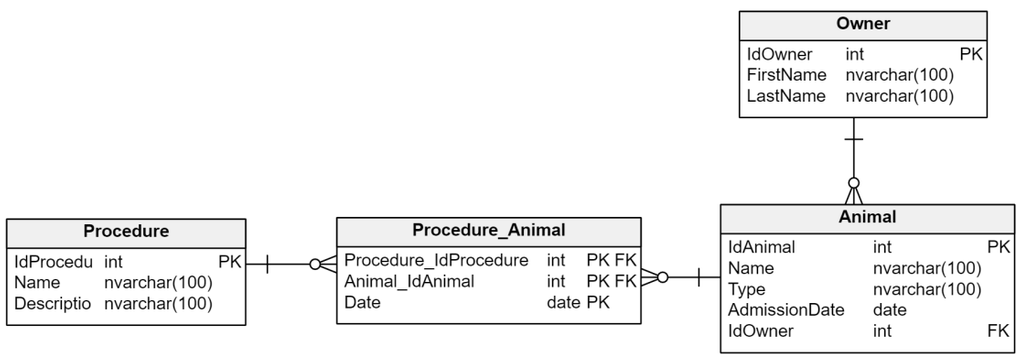
Example of test 1

pgago@pja.edu.pl

21 April 2020

Content of task

Due to the growing needs of the local animal clinic, you have been asked to create a Web API application. There is a DDL script in the database.sql file that creates the database structure, which is shown in the picture below:



- Prepare a controller that will return data about the animals present in the clinic. We would like create an endpoint which will respond to HTTP GET requests. In addition, we want to be able to specify the column by which we sort data and the sorting direction. For this purpose, we pass a parameter in the query string called "sortBy". It can take any value that specifies the name of the column. By default (when we do not specify the value of the sortBy parameter), we sort in descending order of admissionDate column. If an incorrect parameter is given, return a 400 error. The ending should return data in the form: Name, AnimalType, DateOfAdmission, LastNameOfOwner.

  ```
  GET /api/animals?sortBy=name HTTP/1.1
  Host: localhost:51321
  ```

- Prepare endpoint which will allow us to add the animal to the database. It should allow the acceptance of both the animal's data as well as the optional list of treatments it has undergone. In this case, we would like to add both animal and entries to the association table (Procedure_Animal) as part of one transaction. Remember to perform the described activities in one transaction. All fields are required in the database. If the client's request does not contain all the data, we return the code 404.

- Add a middleware that will add an HTTP header with your index number to every server response.

  `IndexNumber: s1234`

- Try to name variables in the correct way.
- Remember to return the correct error codes (400, 404 etc.).
- Remember to handle errors.
- Communication with the database should be in a separate file/service.
- You can use db-mssql or your own local database (if you have one).
- We use the SqlConnection and SqlCommand classes for communication.