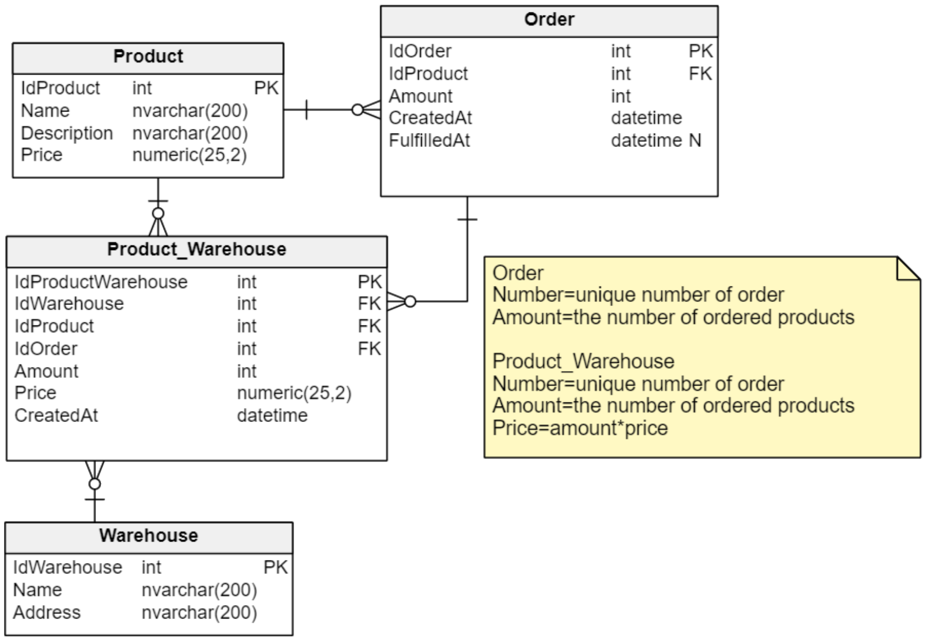
Tutorial 5

In the course of this tutorial, we will again use the SqlConnection and SqlCommand classes. This time around, the logic involved in interacting with our database will be a bit more complex. We create an application for a company that manages the warehouse. The database we use is presented below. In addition, in the **create.sql** file you will find a script that creates tables and fills them with data.



1. Create new REST API application
2. Add controller called **WarehousesController**
3. Inside the controller, add an endpoint that will respond to the following request **HTTP POST send to /api/warehouses**
    a. The endpoint receives the data in the following form:

```
{
    "IdProduct": 1,
    "IdWarehouse": 2,
    "Amount": 20,
    "CreatedAt": "2012-04-23T18:25:43.511Z"
}
```

b. All the fields are required. Amount must be greater than 0.

c. The endpoint should implement the following use case scenario.

Name	Product registration in the warehouse
	Main scenario
1.	We check if the product with the given id exists. Then we check if the warehouse with the given id exists. The amount value passed in the request should be greater than 0.
2.	We can add a product to the warehouse only if there is a product purchase order in the Order table. Therefore, we check if there is a record in the Order table with IdProduct and Amount that matches our request. The CreatedAt of the order should be lower than the CreatedAt in the request.
3.	We check whether this order has been completed by any chance. We check if there is no row with the given IdOrder in the Product_Warehouse table.
4.	We update the FullfilledAt column of the order with the current date and time. (UPDATE)
5.	We insert a record into the Product_Warehouse table. The Price column should corresponds to the price of the product multiplied by amount value from our request. Moreover, we insert the CreatedAt value according to the current time. (INSERT)
6.	As a result of the operation, we return the value of the primary key generated for the record inserted into the Product_Warehouse table.
	Alternative scenario
1a	The product/warehouse with the given id does not exist. We return error 404 with the appropriate message.
2a	There is no suitable order. We return error 404 with the appropriate message.
3a	The order has already been completed. We return the appropriate error code with the message.

4. Then add the **second controller called Warehouses2Controller** with the endpoint corresponding to requests sent to **HTTP POST /api/warehouses2**
    a. The endpoint follows the same logic, but in this case **we execute the stored procedure** (included in the **proc.sql file**).
5. Remember about dependency injection, appropriate names, HTTP return codes
6. Try to use asynchronous and async/await programming methods.