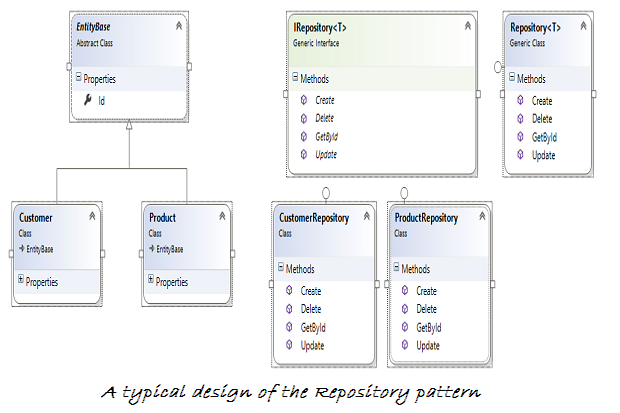
<https://www.infoworld.com/article/3107186/how-to-implement-the-repository-design-pattern-in-c.html>



<https://bool.dev/blog/detail/pattern-repozitoriy-poeaa>

Репозиторий — это посредник между domain слоем и mapping слоем, используя интерфейс, схожий с коллекциями для доступа к объектам области определения.

Стоит отметить, что есть 2 подхода к реализации репозитория, первый из них GenericRepository который выступает посредником достаточного уровня абстракции для всех данных (таблиц базы данных). И репозиторий для каждого типа данных который еще часто реализуется вместе с Unit of Work паттерном.

<https://docs.microsoft.com/en-us/dotnet/architecture/microservices/microservice-ddd-cqrs-patterns/infrastructure-persistence-layer-design>

Repositories are classes or components that encapsulate the logic required to access data sources. They centralize common data access functionality, providing better maintainability and decoupling the infrastructure or technology used to access databases from the domain model layer.

A repository performs the tasks of an intermediary between the domain model layers and data mapping, acting in a similar way to a set of domain objects in memory. Client objects declaratively build queries and send them to the repositories for answers. Conceptually, a repository encapsulates a set of objects stored in the database and operations that can be performed on them, providing a way that is closer to the persistence layer. Repositories, also, support the purpose of separating, clearly and in one direction, the dependency between the work domain and the data allocation or mapping.