Реализация доступа к БД

**1 вариант: применение ADO.Net**

Требуется nuget включить провайдеры данных:

* Microsoft.Data.SqlClient (MSSQL)
* MySql.Data(MySQL)
* Microsoft.Data.Sqlite (SQLite)

**2 вариант: применение ORM**

* C#: Entity Framework Core (EFCore)
* Java: Hubernate
* PHP: Laravel, Yii
* Python: Django, SQLAIchemy

ORM – это объектно-реляционное преобразование, т.е. технология программирования связывающая БД с концепцией ООП через создание виртуальной, объектной базы данных.

**Задачи которые решает ORM:**

* Работа с данными таблиц как с классами
* Работа с классами или объектами как с подходящими для хранения в БД данными
* Избавление от написания SQL-кода
* Упрощение разработки и поддержки кода
* ORM облегчает создание CRUD
  + Create – INSERT
  + Read – SELECT
  + Update – UPDATE
  + Delete – DELETE
* Соответствие(…) на ООП языке
* Таблица – класс
* Столбцы – свойство/поле класса
* Строка – объект
* PK – уникальное свойство объекта
* FK – свойство-ссылка на объект другого класса
* SQL-оператор – LINQ-метод (в C#)

**Подходы применяемые для работы с ORM**

* Code First
  + Сначала создаётся описание классов 🡪 на их основе генерируется БД
* Database First
  + Сначала создаётся БД 🡪на её основе вручную или с помощью mapper-а создаётся описание классов

**3 вариант: применение Micro-ORM Dapper**

* Скачивается провайдер ADO.Net + Dapper
  + Преобразует строки таблицы в объект

**Применение ADO.Net**

ЛЕКЦИИ ПО МДК 01.01!

* Для избавления от SQL –инъекций можно добавить параметры, запрос или команду:

command.Parameters.AddWithValue("@имяПараметра", значениеПараметра);

public static async void MethodName()

{

using SqlConnection connection = new(ConnectionString);

await connection.OpenAsync(); // асинхронный

connection.Open();

// работа с БД

string query = "sql-запрос";

SqlCommand command = new(query, connection);

command.Parameters.AddWithValue("@имяПараметра", значениеПараметра);

// вызов command в асинхрон/синхрон режиме

}

(тип?) – тип допускающий null значение