

Results in a **query object**.

Won‘t be **materialized (QUERY NOT EXECUTED UNTIL DATA NEEDED)**..

I.e. when:

\* Some code will try to enumerate the result, by calling foreach on the collection

The result is converted **ToList()** or ToArray()

We try to access a given value from the result.. using **First/FirstOrDefault/Single/SingleOrDefault()**

**TODO: More on First/Single guys..**

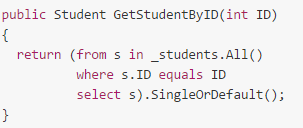
**Accessing record:**

GetStudentbyId..

* **First()** returns the FIRST record from a collection, **THROWS EXCEPTION IF EMPTY**
* **FirstOrDefault()** returns first record from a collection or **NULL (DEFAULT = NULL)** if empty
* **Single()** returns the first record from a collection, and **THROWS EXCEPTION IF EMPTY OR MORE THAN ONE**
* **SingleOrDefault()** returns the first record or **NULL IF COLLECTION IS EMPTY** or **EXCEPTION IF MORE THAN ONE**

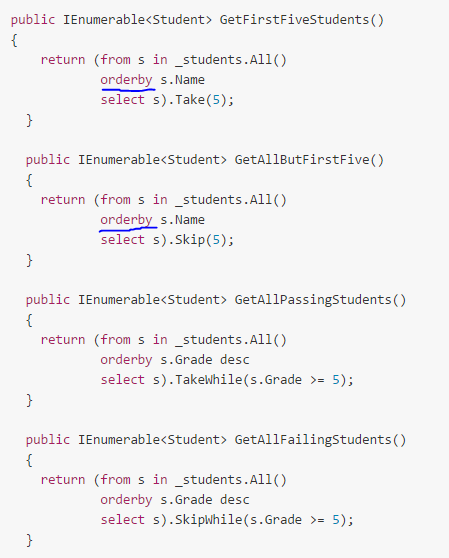
Return \_students.All().SingleOrDefault(s => s.ID == ID);

OR

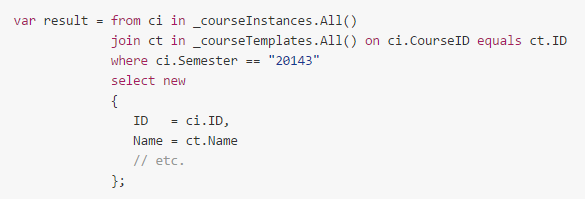


**More than one BUT NOT ALL – Often used with orderby**

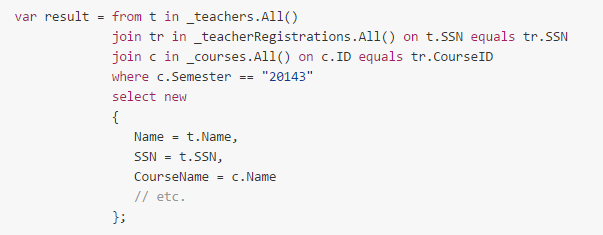
* **Take(int x)** //Returns x first records, **EXCEPTION IF NULL**
* **Skip(int x)** //skips the first x records but returns the rest in the collection.. **EXCEPTION IF NULL**
* **TakeWhile()** //Keeps taking until a SPECIFIC CONDITION is met... and **EXCEPTION if source or predicate is NULL**
* **SkipWhile()** //Skips all elements until a specific condition is met, and **EXCEPTION IF SOURCE OR PREDICATE IS NULL**



Join:



Results in an inner join..



**Other types**

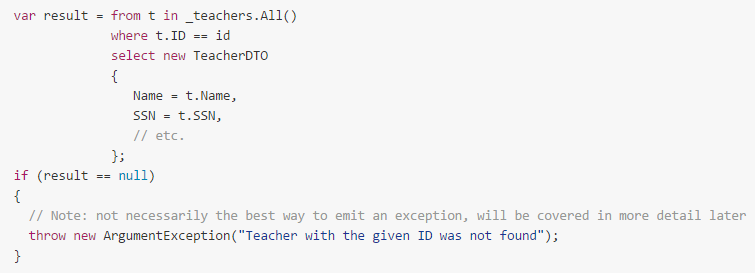
Assume multiple teachers for a given course.. we want a LIST OF ALL COURSES IN A GIVEN SEMESTER.. including the main teacher for EACH course..

**A regular join would skip those course instances where a teacher hasn‘t been registered..**

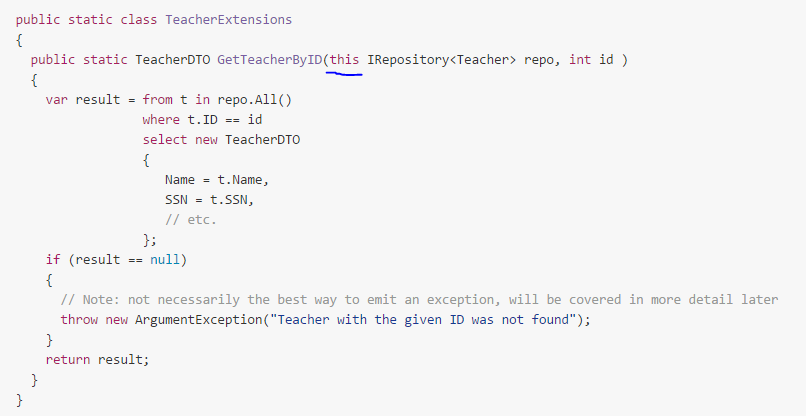
So we would use a **left outer join**... use the **INTO** keyword..

**Extending LINQ**

I.e. when getting something.. and if not exist... throw an exception..



For more in one place.. can create extension method for our repositories:



These methods can be called as if they were member functions in the repos:

Var teacherDTO = \_teachers.GetTeacherById(id);

(must add the appropriate using statement to the files where they should be accessible..)