# Lazy Loading:

One of the important functions of Entity Framework is lazy loading. Lazy loading means delaying the loading of related data, until you specifically request for it. For example, Student class contains StudentAddress as a complex property. So, the context first loads all the students from the database, then it will load the address of a particular student when we access StudentAddress property as shown below.

using (var ctx = new SchoolDBEntities())

{

//Loading students only

IList<Student> studList = ctx.Students.ToList<Student>();

Student std = studList[0];

//Loads Student address for particular Student only (seperate SQL query)

StudentAddress add = std.StudentAddress;

}

However, you can also turn off lazy loading for a particular property or an entire context. To turn off lazy loading for a particular property, do not make it virtual. To turn off lazy loading for all entities in the context, set its configuration property to false:

using System;

using System.Data.Entity;

using System.Data.Entity.Infrastructure;

using System.Data.Entity.Core.Objects;

using System.Linq;

public partial class SchoolDBEntities : DbContext

{

public SchoolDBEntities(): base("name=SchoolDBEntities")

{

this.Configuration.LazyLoadingEnabled = false;

}

protected override void OnModelCreating(DbModelBuilder modelBuilder)

{

throw new UnintentionalCodeFirstException();

}

}

**Rules for lazy loading:**

1. *context.Configuration.ProxyCreationEnabled* should be true.
2. *context.Configuration.LazyLoadingEnabled* should be true.
3. Navigation property should be defined as public, virtual. Context will **NOT** do lazy loading if the property is not defined as virtual

# Eager Loading:

Eager loading is the process whereby a query for one type of entity also loads related entities as part of the query. Eager loading is achieved using the **Include()** method.

In the following example, it gets all the students from the database along with it's standards using Include() method.

**LINQ Query Syntax:**

using (var context = new SchoolDBEntities())

{

var res = (from s in context.Students.Include("Standard")

where s.StudentName == "Student1"

select s).FirstOrDefault<Student>();

}