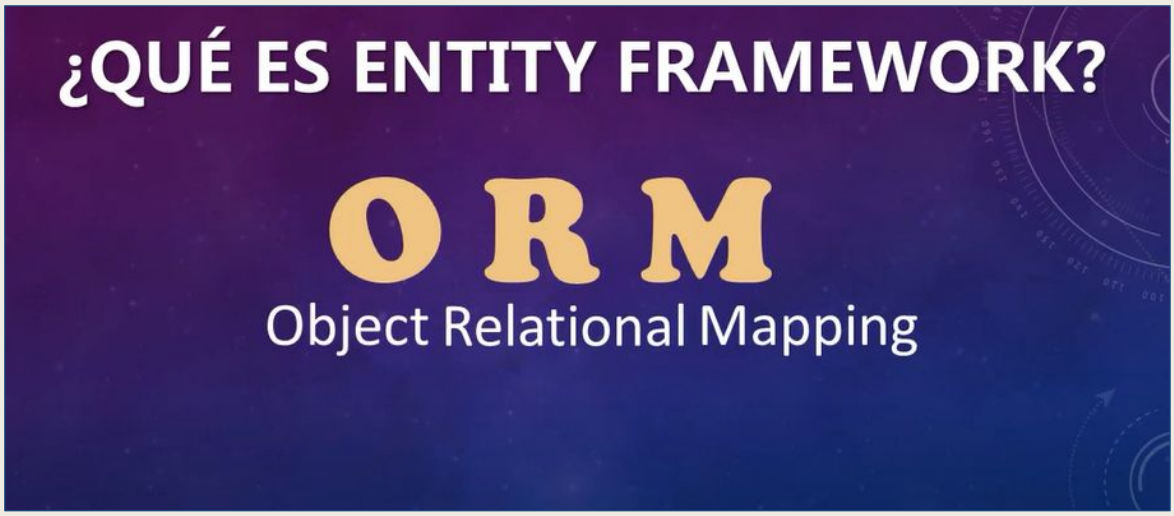
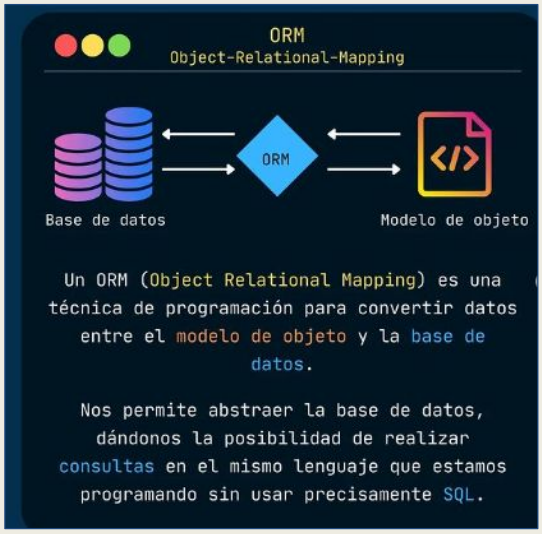
Acceso a datos mediante EF

Entity Framework

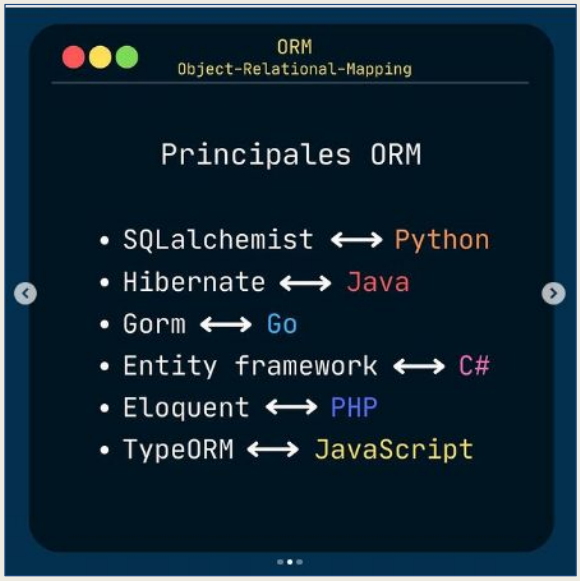


ORM - Object Relational Mapping



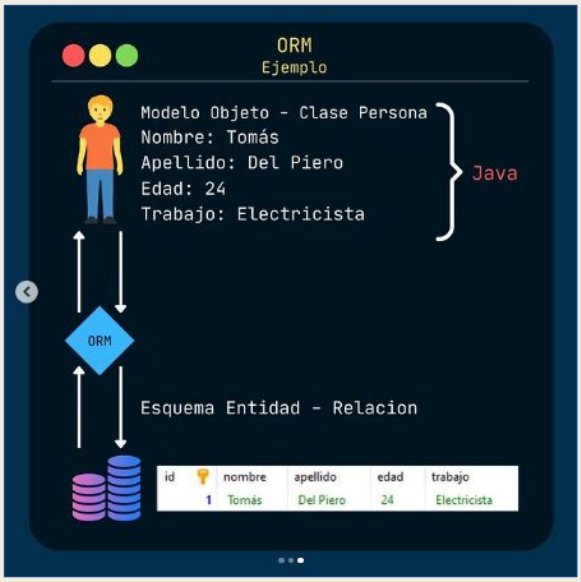
¿Qué es? ¿Para qué se usa?

ORM - Object Relational Mapping



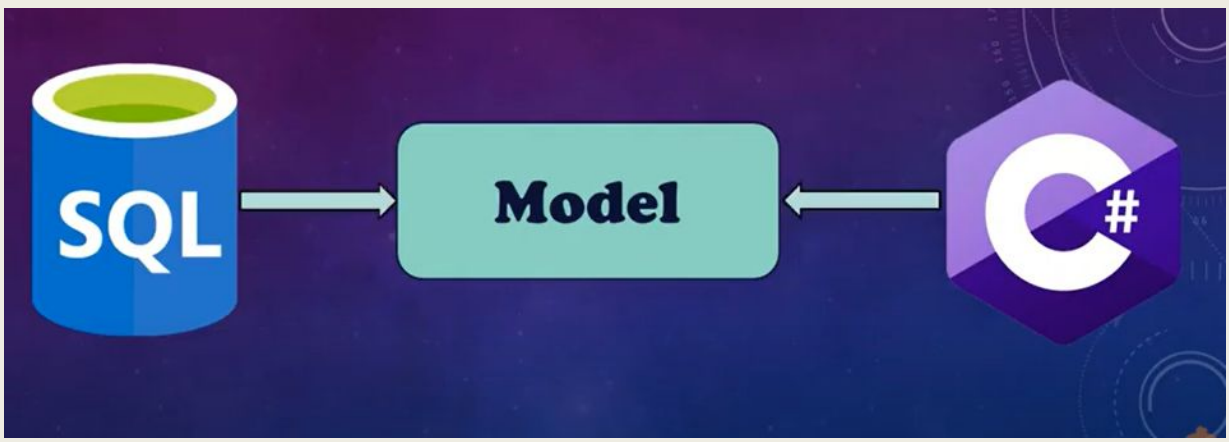
Ejemplos de ORM de distintos lenguajes

ORM - Object Relational Mapping



ORM - Object Relational Mapping

- Técnica para convertir datos entre sistemas que son en principio incompatibles



Entity Framework

- ORM de código abierto exclusivo para aplicaciones .net
- Permite a los desarrolladores:
  - trabajar con **datos** en forma de **propiedades** y **objetos** específicos del dominio. *Ejem. trabajaremos con "Clientes" y "Pedidos", sin tener que preocuparnos por las tablas y columnas de la base de datos subyacente que almacena dichos datos.*
  - **reducir el código** específico de acceso a datos

¿Qué debo usar EF6 o EF Core?

- EF Core es una versión más moderna, ligera y extensible de Entity Framework con capacidades y ventajas muy similares a EF6.
- EF Core es una "reescritura completa". Contiene **características nuevas** que no están disponibles en EF6, sin embargo todavía **carece** de algunas de las **funcionalidades más avanzadas** de asignación de EF6.
- *Recomendaciones para el proceso de selección*

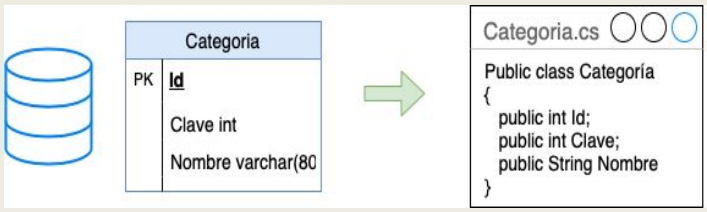
Entity Framework

EF permite trabajar con dos **flujos/modos de trabajo**:

- **Code-First:** permite definir el modelo mediante clases C#

  escenario 1(la bases de datos existe) y escenario 2 (la bases de datos no existe)

- **Database-First:** Crea el modelo a partir de una base de datos.



Entity Framework

- **Code-First:** Creas tu clase y mediante un comando de .Net te crea la tabla, sus llaves, índices, etc. en la base de datos de tu elección. *(* Este enfoque puede usarse para establecer como destino una base de datos existente o para crear una nueva base de datos *)*

- **Database-First:** Si ya tienes tu BD, con un comando de .Net te crea las clases para el módelo, índices, llaves foráneas, etc.

DbContext Class *(o clase de contexto)*

- Es la clave del funcionamiento de EF *(la clase más importante mientras se trabaja con EF 6 o EF Core)*
- Deriva de System.Data.Entity.DbContextDbContext *(en EF 6 y EF Core)*
- Representa una sesión con la base de datos subyacente mediante la cual puede realizar operaciones CRUD *(Cretate, Read, Update, Delete).*

DbContext Class

- Responsable de las siguientes actividades:
  - **Consulta:** *Convierte consultas LINQ-to-Entities en consultas SQL y las envía a la base de datos.*
  - **Seguimiento de cambios**: *realiza un seguimiento de los cambios que ocurrieron en las entidades después de consultar desde la base de datos.*
  - **Persistencia de los datos:** *realiza las operaciones Insertar, Actualizar y Eliminar en la base de datos, en función de los estados de la entidad.*
  - **Almacenamiento en caché:** *proporciona almacenamiento en caché de primer nivel de forma predeterminada. Almacena las entidades que se han recuperado durante el tiempo de vida de una clase de contexto.*
  - *…*

DbContext Class

- *Responsable de…(cont.):*
  - **Gestionar relaciones:** *gestiona relaciones utilizando CSDL, MSL y SSDL en el enfoque Db-First o Model-First, y utilizando configuraciones de API fluidas en el enfoque Code-First.*
  - **Materialización de objetos:** *Convierte datos sin procesar de la base de datos en objetos de entidad.*

DbContext Class - ejemplo

```
using System.Data.Entity;

public class SchoolContext : DbContext
{
    public SchoolContext()
    {

    }
    // Entities
    public DbSet<Student> Students { get; set; }
    public DbSet<StudentAddress> StudentAddresses { get; set; }
    public DbSet<Grade> Grades { get; set; }
}
```

*La clase **SchoolContext** deriva de DbContext, lo que la convierte en una **clase de contexto**. También incluye un conjunto de entidades para las **entidades** <**Student**>, <**StudentAddress**> y <**Grade**>*

EF 6 - Consulta de entidades -

- tipos de sintaxis disponibles:
  - sintaxis de expresiones de consulta
  - sintaxis de consultas basadas en métodos *(expresiones lambda)*

EF 6 - Consulta de entidades -

Consulta **LINQ** en la base de datos *(técnica1)*

- crea la consulta en la base de datos, lo que **siempre** conlleva un viaje de ida y vuelta a la **base de datos** aún cuando las entidades devueltas ya existan en el contexto.

Método **Find** de DbSet *(técnica2)*

- usa el valor de **clave principal** para intentar buscar una entidad cuyo seguimiento realiza el contexto
- diferencias importantes:
  - Solo se realiza el viaje de ida y vuelta a la base de datos si la entidad con la clave especificada no se encuentra en el contexto.
  - Find devuelve entidades que están en estado Added. Es decir, Find devuelve entidades que se han agregado al contexto pero que aún no se han guardado en la base de datos.

EF 6 - Consulta de entidades - Linq

```
using (var context = new BloggingContext())
{
    // Consulta de todos los blogs con nombres que comienzan con B
    var blogs = from b in context.Blogs
                where b.Name.StartsWith("B")
                select b;

    // Consulta para el Blog llamado ADO.NET Blog
    var blog = context.Blogs
                .Where(b => b.Name == "ADO.NET Blog")
                .FirstOrDefault();
}
```

EF 6 - Consulta de entidades - Linq

La consulta se ejecuta en la base de datos cuando:

- se enumera mediante una instrucción foreach *(C#)* o For Each *(Visual Basic)*
- se enumera mediante una operación de recopilación como ToArray, ToDictionary o **ToList**
- se llama a los métodos siguientes: el método de extensión Load en DbSet, DbEntityEntry.Reload y Database.ExecuteSqlCommand
- los operadores LINQ, como **First** o Any, se especifican en la parte más externa de la consulta

EF 6 - Consulta de entidades - Find()

El **método Find** de DbSet usa el valor de la **clave principal.**

Si la entidad no se encuentra en el contexto ni en la base de datos, se devuelve null.

```
using (var context = new BloggingContext())
{
     var blog = context.Blogs.Find(3);

    // busca un usuario que tenga una clave principal de cadena
    var user = context.Users.Find("johndoe1987");
}
```

EF 6 - Consulta de entidades relacionadas