Accessing a Database

Introduction to the Entity Framework (EF)

- The Entity Framework provides:
  - Database access using objects and LINQ
- The Entity Framework supports:
  - Reading, adding, updating, deleting data
  - Various database types
  - All usable using the same programming model

EF Working Parts

Entity class Product

```
int Id {get; set;}
string Name {get; set;}
```

Entity class Sale

```
int Id {get; set;}
DateTime Date {get; set;}
```

DbContext

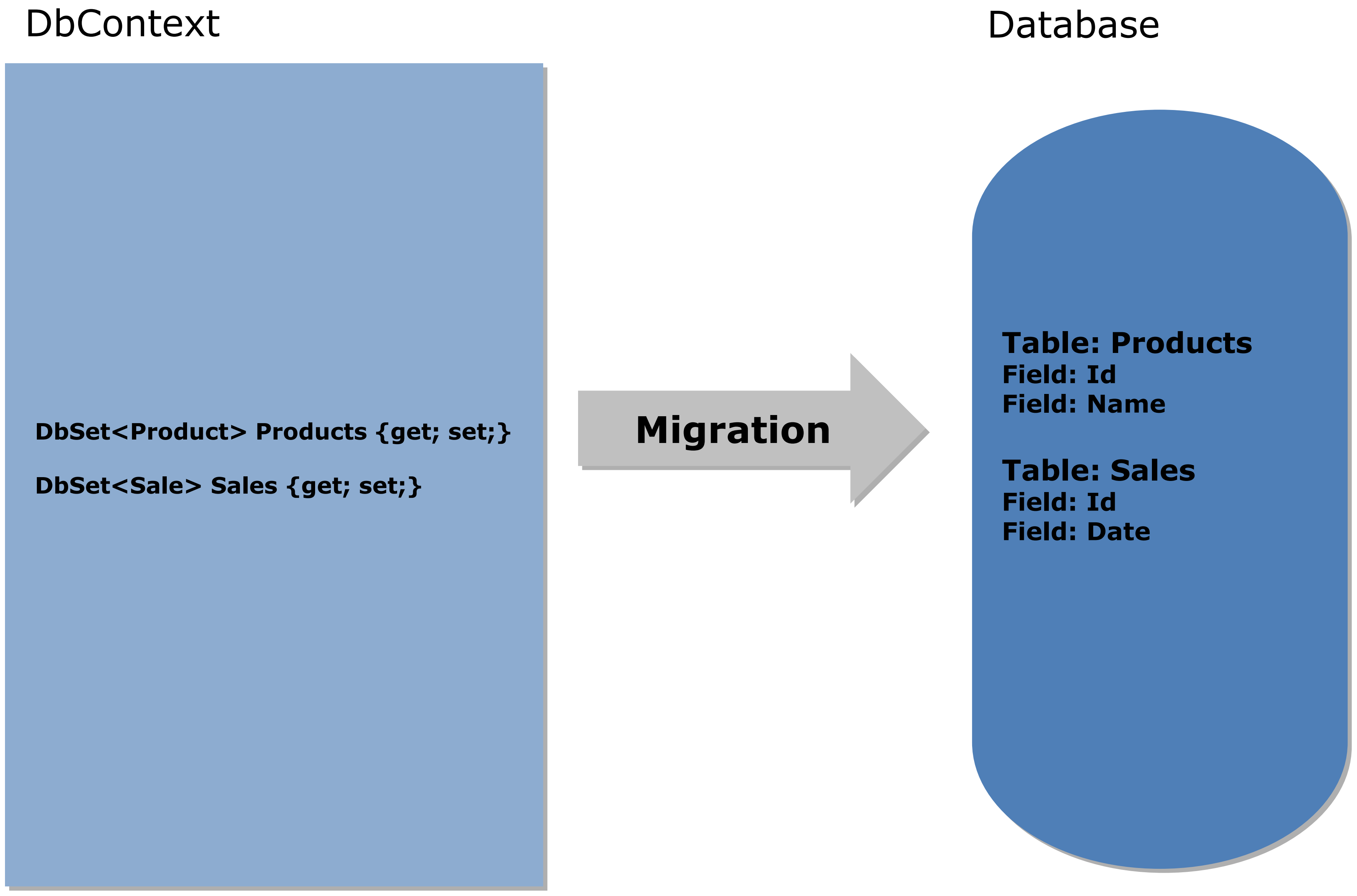
```
DbSet<Product> Products {get; set;}

DbSet<Sale> Sales {get; set;}
```

Using the EF Tools

- Tools are needed to:
  - Manage migrations
  - Update the database

EF Model to Database



Reading and Modifying Data by Using the Entity Framework

- Reading data

```
FourthCoffeeEntities DBContext = new FourthCoffeeEntities();

// Print a list of employees.
foreach (FourthCoffee.Employees.Employee emp in
DBContext.Employees)
{
    Console.WriteLine("{0} {1}", emp.FirstName, emp.LastName);
}
```

- Modifying data

```
var emp = DBContext.Employees.First(e => e.LastName == "Prescott");
if (emp != null)
{
    emp.LastName = "Forsyth";
    DBContext.SaveChanges();
}
```

Forcing Query Execution

- Deferred query execution—default behavior for most queries
- Immediate query execution—default behavior for queries that return a singleton value
- Forced query execution—overrides deferred query execution:
  - **ToArray**
  - **ToDictionary**

```
IList<Employee> emp = (from e in FCEntities.Employees
                       orderby e.LastName
                       select e).ToList();
```

Lab