MiniORM Core

Custom ORM Framework

Overview and Features

MiniORM Core: Overview

- Designed after **Entity Framework Core**
- Provides **LINQ-based data queries** and **CRUD operations**
- **Change tracking** of in-memory objects
- Maps navigation properties
- Maps collections
  - **One-to-many**, **Many-to-many**, etc.

MiniORM Core Workflow: Overview

- **Define data model** (database-first)
  - **Entity** Classes
  - **`DbContext`** (with **`DbSets`**)
- Initialize **`DbContext`**
  - Using connection string
- Query data using context
- Manipulate data (add/remove/update entities)
- Context gets persisted into database

MiniORM Components (1)

- The **DbContext** class
  - Holds the **database connection** and the **DB Sets**
  - Provides **LINQ-based** data access
  - Provides **change tracking**, and an API for **CRUD** operations
- **DBSets**
  - Hold **entities** (objects with their attributes and relations)
  - Each database **table** is typically mapped to a single **C# class**

MiniORM Components (2)

- **Associations** (relationship mappings)
  - An association is **a primary key** / **foreign key**-based **relationship** between two entity classes
  - Allows **navigation** from one entity to another

```
var courses = student.Courses.Where(…);
```

  - MiniORM **supports one-to-one**, **one-to-many** and **many-to-many** relationships

ChangeTracker: Cloning Entities (MiniORM)

- In order to check for entity modification, the change tracker **clones** all entities on initialization
- Cloning process
  - Create **new** blank **instance** of entity
  - Find all **properties**, which are valid SQL types
  - Set blank instance's property **values** to existing entity values