



1. **Что такое ORM?**

**ORM** или **Object-relational mapping** — это технология программирования, которая позволяет преобразовывать несовместимые типы моделей в ООП, в частности, между хранилищем данных и объектами программирования. ORM используется для упрощения процесса сохранения объектов в реляционную базу данных и их извлечения, при этом ORM сама заботится о преобразовании данных между двумя несовместимыми состояниями

1. **Зачем используются ORM-решения при работе с базами данных? Опишите их достоинства и недостатки.**

Использование ORM в проекте избавляет разработчика от необходимости работы с SQL и написания большого количества кода, часто однообразного и подверженного ошибкам. Весь генерируемый ORM код предположительно хорошо проверен и оптимизирован, поэтому не нужно в целом задумывается о его тестировании. Это несомненно является плюсом, но в тоже время не стоит забывать и о минусах.

Основной из них — это потеря производительности. Это происходит потому, что большинство ORM предназначены для обработки широкого спектра сценариев использования данных, гораздо большего, чем любое отдельное приложение когда-либо сможет использовать.

Вопрос о целесообразности использования ORM по большому счету затрагивается только в больших проектах, которые сталкиваются с высокой нагрузкой, здесь приходится выбирать что более приоритетно — удобство или производительность? Конечно, работа с БД посредством грамотно написанного SQL-кода будет намного эффективнее, но не стоит забывать и о таком параметре, как время — то, что с легкостью пишется с использованием ORM за неделю, можно реализовывать ни один месяц собственными усилиями. Кроме того, большинство современных ORM позволяют программисту при необходимости самому задавать код SQL-запросов. Без сомнений, для небольших проектов использование ORM будет куда более оправдано, чем разработка собственных библиотек для работы с БД.

1. **Что такое Entity Framework?**

**Entity Framework** представляет специальную объектно-ориентированную технологию на базе фреймворка .NET для работы с данными. Если традиционные средства **ADO.NET** позволяют создавать подключения, команды и прочие объекты для взаимодействия с базами данных, то **Entity Framework** представляет собой более высокий уровень абстракции, который позволяет абстрагироваться от самой базы данных и работать с данными независимо от типа хранилища. Если на физическом уровне мы оперируем таблицами, индексами, первичными и внешними ключами, но на концептуальном уровне, который нам предлагает Entity Framework, мы уже работает с объектами.

1. **Что такое EDM в Entity Framework и из каких уровней она состоит?**

**Entity Data Model**

Эта модель сопоставляет классы сущностей с реальными таблицами в БД.

Entity Data Model состоит из трех уровней: концептуального, уровень хранилища и уровень сопоставления (маппинга).

На концептуальном уровне происходит определение классов сущностей, используемых в приложении.

Уровень хранилища определяет таблицы, столбцы, отношения между таблицами и типы данных, с которыми сопоставляется используемая база данных.

Уровень сопоставления (маппинга) служит посредником между предыдущими двумя, определяя сопоставление между свойствами класса сущности и столбцами таблиц.

1. Перечислите способы создания моделей. Опишите их действия.
2. **Назовите основные отличия EF от других технологий работы с базами данных.**

Если традиционные средства **ADO.NET** позволяют создавать подключения, команды и прочие объекты для взаимодействия с базами данных, то **Entity Framework** представляет собой более высокий уровень абстракции, который позволяет абстрагироваться от самой базы данных и работать с данными независимо от типа хранилища.

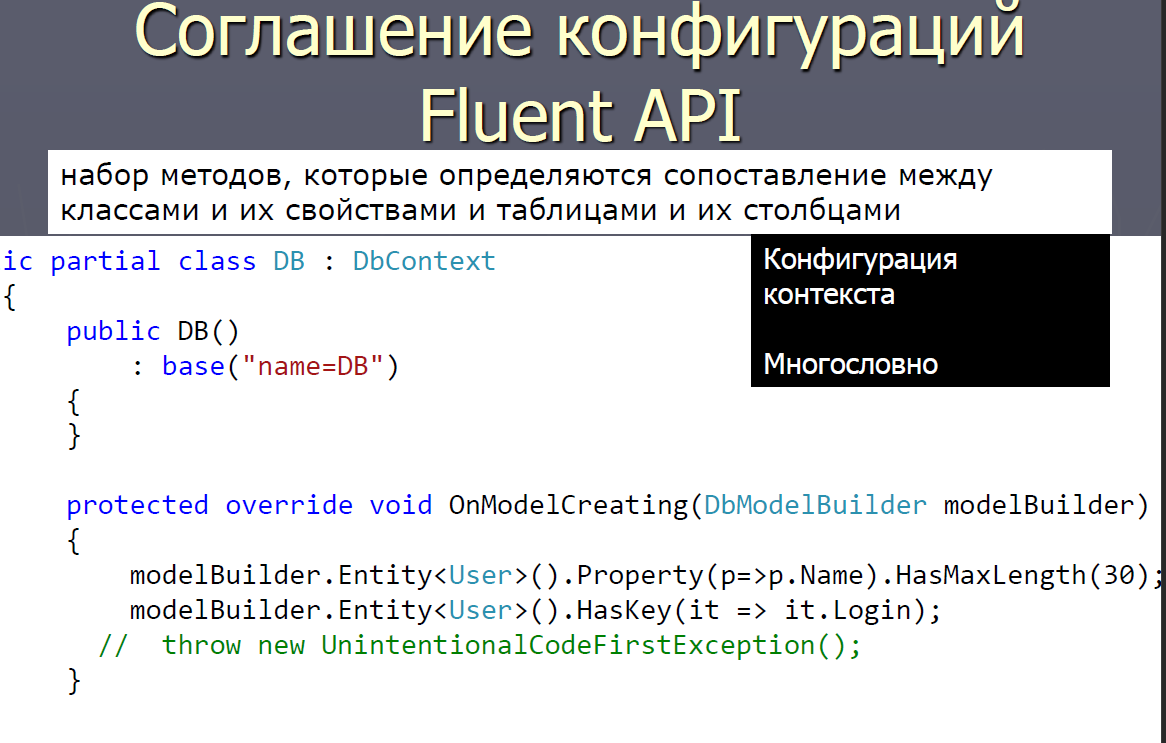
1. **Для чего и как используется DbContext?**

Является базовым классом Entity Framework и предоставляет широкие возможности по работе с базой данных: создание запросов, отслеживание изменений и сохранение данных в базе.

1. **Как и для чего используются аннотации при настройке конфигураций Code First?**

Аннотации представляют настройку сопоставления моделей и таблиц с помощью атрибутов. Большинство классов аннотаций располагаются в пространстве System.ComponentModel.DataAnnotations, которое нам надо подключить в файл c# перед использованием аннотаций.

1. **Как и для чего используются Fluent API при настройке конфигураций Code First?**



1. **Какие существуют способы загрузки связанных данных?**

В Entity Framework есть три способа загрузки данных:

* eager loading("жадная загрузка")
* explicit loading("явная загрузка")
* lazy loading("ленивая загрузка")

**Отложенная загрузка (lazy loading)** заключается в том, что Entity Framework автоматически загружает данные, при этом не загружая связанные данные. Когда потребуются связанные данные Entity Framework создаст еще один запрос к базе данных.

**Прямая загрузка данных (eager loading)** позволяет указать в запросе какие связанные данные нужно загрузить при выполнении запроса. Благодаря этому, когда в коде вы будете ссылаться на связанную таблицу через навигационное свойство, SQL-запрос не будет направляться в базу данных, т.к. связанные данные уже будут загружены при первом запросе.

Последним вариантом загрузки данных в Entity Framework является **явная загрузка (explicit loading)** данных. Явная загрузка, как и отложенная загрузка, не приводит к загрузке всех связанных данных в первом запросе. Но при этом, в отличие от отложенной загрузки, при вызове навигационного свойства связанного класса, эта загрузка не приводит к автоматическому извлечению связанных данных, вы должны явно вызвать метод Load(), если хотите загрузить связанные данные.

1. **В чем разница между концепциями Code First и Database First?**

**Code-First** – новая возможность, стоящая в одном ряду с Model-First и Database-First. Теперь можно сначала написать код, описывающий классы-модели, а потом фреймворк автоматически создаст БД по такому коду.  
Самое приятное, что виды отношений будут распознаны – достаточно определить просто ссылки на объекты для 1:1, ICollection для отношения 1:n, и взаимные ICollection для m:n. В этом случае промежуточная таблица также создастся автоматически. Для обеспечения «ленивой» загрузки хватит ключевого слова virtual в определении свойства.

Entity Framework предполагает три возможных способа взаимодействия с базой данных:

* **Database first**: Entity Framework создает набор классов, которые отражают модель конкретной базы данных
* **Model first**: сначала разработчик создает модель базы данных, по которой затем Entity Framework создает реальную базу данных на сервере.
* **Code first**: разработчик создает класс модели данных, которые будут храниться в бд, а затем Entity Framework по этой модели генерирует базу данных и ее таблицы

1. **Зачем EF использует миграции? Как создать новую миграцию?**

Миграции позоляют вносить изменения в базу данных при изменениях моделей и контекста данных.

**13. Расскажите алгоритм выполнения CRUD-запросов при работе с БД через библиотеку Entity Framework.**

Базовым подходом к написанию запросов в Entity Framework является использование расширения языка C# - **LINQ**, которое предоставляет набор методов для работы с коллекциями.