1. **What is Angular and why is it used?**

   Answer:
   Angular is a platform and framework for building single-page client applications using HTML and TypeScript. It provides a comprehensive solution for building dynamic, responsive, and scalable web applications. It's used because it provides tools for routing, forms management, HTTP client, and more, making it a powerful choice for building modern web applications.

2. **Explain two-way data binding in Angular.**

   Answer:
   Two-way data binding is a mechanism in Angular that allows automatic synchronization of data between the model (component) and the view (template). This means that if data changes in the model, it reflects in the view, and if the user updates the view, the model is updated as well. This is achieved using the `ngModel` directive.

   Example:

   ```
   <input [(ngModel)]="name">
   ```

   Here, any change to `name` in the component will update the input field, and vice versa.

3. **What is the difference between `ngOnInit` and `constructor` in Angular?**

   Answer:
   The `constructor` is used to initialize the class and inject dependencies. It is called before Angular initializes the component. However, `ngOnInit` is a lifecycle hook that is called after Angular has initialized all the data-bound properties of a component.

   - `constructor`: Used for dependency injection.
   - `ngOnInit`: Ideal for fetching data or performing logic once the component has been initialized.

4. **What is a service in Angular, and how is it different from a component?**

   Answer:
   A service in Angular is a class that provides functionality to other components and services. It typically contains business logic and data management, allowing code reusability. Components in Angular are used to manage the UI and user interactions, while services handle data and logic that are independent of the UI.

   - Components: Handle view logic.
   - Services: Handle business logic and data.

5. **What is Angular CLI and how is it useful?**

   Answer:
   Angular CLI (Command Line Interface) is a tool that helps automate the workflow of Angular applications. It provides commands to generate components, services, modules, and other Angular entities, build the application, run tests, and more. It

simplifies tasks such as creating new projects, adding features, running a development server, and performing builds.

.NET Interview Questions

1. **What is the difference between .NET Framework and .NET Core?**

   Answer:
   The .NET Framework is a Windows-only platform for building web, desktop, and console applications. .NET Core is a cross-platform version of .NET that runs on Windows, Linux, and macOS. .NET Core is more lightweight, modular, and optimized for modern cloud-based applications. .NET 5 (and later versions) are the evolution of .NET Core, making it a unified platform.

2. **Explain the concept of Dependency Injection in .NET Core.**

   Answer:
   Dependency Injection (DI) is a design pattern used to implement Inversion of Control (IoC). In .NET Core, DI is built into the framework, and services can be injected into constructors, methods, or properties to improve modularity and testability. It allows classes to be decoupled from their dependencies, making the system more maintainable.

   - Services are registered in the `ConfigureServices` method of `Startup.cs`.
   - Dependencies are injected into constructors of classes.

3. **What are the differences between `IEnumerable` and `IQueryable` in .NET?**

   Answer:

   - `IEnumerable`: It is used for querying in-memory collections like arrays, lists, etc. It performs operations in memory, and the query is executed immediately.
   - `IQueryable`: It is used for querying data from an external source (e.g., a database) and supports deferred execution. It allows LINQ queries to be translated into SQL queries that can be executed by the database.

4. **What is the purpose of middleware in .NET Core?**

   Answer:
   Middleware in .NET Core is a piece of software that is assembled into an application pipeline to handle requests and responses. Each middleware component can perform operations before passing control to the next component in the pipeline. Common examples include authentication, logging, error handling, and routing.

5. **What are async and await in C#?**

   Answer:
   `async` and `await` are used to create asynchronous operations in C#.

- `async`: Marks a method as asynchronous, allowing it to run in a non-blocking way.
- `await`: Pauses the execution of an asynchronous method until the awaited task completes.

Example:

```
public async Task<int> GetDataAsync()
{
    var data = await GetDataFromDatabaseAsync();
    return data;
}
```

In this example, `GetDataFromDatabaseAsync()` is awaited, meaning the method does not block the thread while waiting for the data to be fetched.

6. **What is the difference between `abstract class` and `interface` in C#?**

   Answer:

   - `Abstract Class`: Can have both abstract (without implementation) and concrete (with implementation) methods. It can have fields, properties, and constructors. A class can inherit only one abstract class.
   - `Interface`: Defines a contract that implementing classes must follow, but it cannot have any implementation. A class can implement multiple interfaces.
7. **What are Entity Framework migrations and how do they work?**

   Answer:
   Entity Framework (EF) migrations allow you to update and manage the schema of your database as your application's data model changes. Migrations track changes in your code (like adding or modifying tables) and apply those changes to the database in a controlled manner.

   - `Add-Migration`: Generates a migration file with changes.
   - `Update-Database`: Applies the migration to the database.

   Example:

   ```
   Add-Migration AddNewColumn
   Update-Database
   ```

1. **What is Angular?**
   Answer: A TypeScript-based framework for building web applications.
2. **What is TypeScript?**
   Answer: A superset of JavaScript with static typing.
3. **What is a component in Angular?**
   Answer: A building block of the UI in Angular.
4. **What is a directive?**
   Answer: A class that modifies the behavior of elements in the DOM.
5. **What is a module in Angular?**
   Answer: A container for organizing related components, services, etc.

6. **What is Dependency Injection (DI)?**
   Answer: A design pattern for providing dependencies to a class.
7. **What is a service in Angular?**
   Answer: A class for business logic and data management.
8. **What does `ngOnInit` do?**
   Answer: Initializes data when the component is created.
9. **What is RxJS?**
   Answer: A library for reactive programming using observables.
10. **What is the purpose of `ngModel`?**
    Answer: Two-way data binding between form inputs and components.

.NET Interview Questions

1. **What is .NET Core?**
   Answer: A cross-platform, open-source version of .NET.
2. **What is the CLR?**
   Answer: Common Language Runtime, the execution engine for .NET applications.
3. **What is an interface?**
   Answer: A contract that defines a set of methods without implementation.
4. **What is a delegate?**
   Answer: A type that references a method with a specific signature.
5. **What is LINQ?**
   Answer: Language Integrated Query for querying collections.
6. **What is an abstract class?**
   Answer: A class that cannot be instantiated and can contain abstract methods.
7. **What is Entity Framework?**
   Answer: An ORM (Object-Relational Mapper) for working with databases in .NET.
8. **What is `async` and `await`?**
   Answer: Keywords for asynchronous programming.
9. **What is a constructor?**
   Answer: A method used to initialize an object.
10. **What is a static class?**
    Answer: A class that cannot be instantiated and can only contain static members.

Angular - Import Related Questions

1. **How do you import a module in Angular?**
   Answer:
   Using the `import` keyword:

   ```
   import { NgModule } from '@angular/core';
   ```

2. **How do you import a service into a component?**
   Answer:
   Using the `import` keyword:

   ```
   import { MyService } from './my-service.service';
   ```

3. **What is the purpose of the `@NgModule` decorator in Angular**
   Answer:
   It marks a class as an Angular module and imports other modules, components, pipes, and service.
4. **How do you import external libraries in Angular?**
   Answer:
   You can install via `npm` and then import in the component or module:

   ```
   npm install lodash
   import * as _ from 'lodash';
   ```

5. **How do you import routing module in Angular?**
   Answer:
   Import `RouterModule` in the main application module:

   ```
   import { RouterModule } from '@angular/router';
   ```

.NET - Import Related Questions

1. **How do you import namespaces in C#?**
   Answer:
   Using the `using` directive:

   ```
   using System;
   using System.Linq;
   ```

2. **How do you import a class from another namespace in .NET?**
   Answer:
   Using the `using` keyword to reference a namespace:

   ```
   using MyApp.Models;
   ```

3. **What is the purpose of the `using` statement in C#?**
   Answer:
   It allows you to import namespaces and resources, simplifying access to classes.
4. **How do you import an assembly in .NET?**
   Answer:
   By adding a reference to the assembly in the project and using `using` to access its classes.
5. **How do you use external libraries in .NET?**
   Answer:
   Using NuGet to install the library and then importing it:

   ```
   Install-Package Newtonsoft.Json
   using Newtonsoft.Json;
   ```

Entity Framework - Import Related Questions

1. **How do you import the Entity Framework core in a project?**
   Answer:
   By installing the `Microsoft.EntityFrameworkCore` package via NuGet and importing the necessary namespaces:

   ```
   Install-Package Microsoft.EntityFrameworkCore
   using Microsoft.EntityFrameworkCore;
   ```

2. **How do you import the `DbContext` class in Entity Framework?**
   Answer:
   By importing `Microsoft.EntityFrameworkCore`:

   ```
   using Microsoft.EntityFrameworkCore;
   ```

3. **How do you import models in Entity Framework?**
   Answer:
   Simply by referencing the class that represents the model:

   ```
   using MyApp.Models;
   ```

4. **How do you import and use `DbSet` in Entity Framework?**
   Answer:
   A `DbSet` is imported as part of your `DbContext` class:

   ```
   public DbSet<Customer> Customers { get; set; }
   ```

5. **How do you import a connection string for Entity Framework?**
   Answer:
   The connection string is added in `appsettings.json`, and imported in `DbContext`:

   ```
   "ConnectionStrings": {
     "DefaultConnection": "YourConnectionStringHere"
   }
   ```

   In `DbContext`:

   ```
   optionsBuilder.UseSqlServer(Configuration.GetConnectionString("Defaul
   tConnection"));
   ```

SQL

10 Important SQL Questions with Answers (1-Year Experience)

1. **What is SQL?**
   - **Answer:** SQL (Structured Query Language) is used for querying and managing relational databases.
2. **What is a Primary Key?**
   - **Answer:** A primary key uniquely identifies each record in a table and cannot have NULL values.
3. **What is a Foreign Key?**
   - **Answer:** A foreign key is a column that links to the primary key of another table, ensuring referential integrity.
4. **What is the difference between `INNER JOIN` and `LEFT JOIN`?**
   - **Answer:**
     - **INNER JOIN** returns only matching rows from both tables.
     - **LEFT JOIN** returns all rows from the left table and matching rows from the right table (NULL if no match).
5. **What is normalization in SQL?**
   - **Answer:** Normalization is the process of organizing data to reduce redundancy and dependency by dividing a database into smaller, related tables.
6. **What is the purpose of the `GROUP BY` clause?**
   - **Answer:** The `GROUP BY` clause groups rows that share a property, often used with aggregate functions like `COUNT()`, `SUM()`, and `AVG()`.
7. **What is a `JOIN`?**
   - **Answer:** A JOIN is used to combine rows from two or more tables based on a related column between them.
8. **What is the `HAVING` clause used for?**
   - **Answer:** The `HAVING` clause is used to filter the result set after `GROUP BY` has been applied.
9. **What is an `INDEX`?**
   - **Answer:** An index is a performance-tuning method to speed up the retrieval of rows from a database table.
10. **What is a `VIEW` in SQL?**
    - **Answer:** A view is a virtual table based on the result of a query. It simplifies complex queries and provides security by restricting access to certain data.

---

10 Short Answer SQL Questions (1-Year Experience)

1. **What is a `NULL` value in SQL?**
   - **Answer:** A `NULL` value represents missing or unknown data.
2. **What is the `DISTINCT` keyword used for?**
   - **Answer:** The `DISTINCT` keyword removes duplicate rows from the result set.
3. **What is a `Subquery`?**
   - **Answer:** A subquery is a query nested inside another query to retrieve data based on the result of the outer query.
4. **What is the `COUNT()` function?**
   - **Answer:** `COUNT()` is an aggregate function that returns the number of rows that match a specified condition.
5. **What is the difference between `WHERE` and `HAVING`?**
   - **Answer:** `WHERE` filters rows before grouping, while `HAVING` filters groups after the `GROUP BY` clause.
6. **What is a `Trigger` in SQL?**
   - **Answer:** A trigger is a database object that automatically executes a specified action when certain events occur (e.g., insert, update, delete).
7. **What does `LIMIT` do in SQL?**
   - **Answer:** The `LIMIT` clause restricts the number of rows returned by a query.
8. **What is a `UNION`?**
   - **Answer:** `UNION` combines the result sets of two or more queries and removes duplicates.
9. **What is the `ALTER` statement used for?**
   - **Answer:** `ALTER` is used to modify the structure of an existing database object (e.g., table, column).
10. **What is the `IN` operator in SQL?**
    - **Answer:** The `IN` operator is used to check if a value matches any value in a list or subquery.

---

10 Query-Based SQL Questions (1-Year Experience)

1. **Write a query to get the total number of employees in each department.**
   - **Answer:**

```
SELECT department_id, COUNT(*) AS total_employees
FROM employees
GROUP BY department_id;
```

2. **Write a query to find the second-highest salary from an employee table.**
   - **Answer:**

```
SELECT MAX(salary)
FROM employees
WHERE salary < (SELECT MAX(salary) FROM employees);
```

3. **Write a query to retrieve all customers who have placed an order.**
   - **Answer:**

```
SELECT DISTINCT customer_id
FROM orders;
```

4. **Write a query to find employees with the same salary.**
   - **Answer:**

```
SELECT salary, COUNT(*)
FROM employees
GROUP BY salary
HAVING COUNT(*) > 1;
```

5. **Write a query to get the names of employees who joined after January 1st, 2020.**
   - **Answer:**

```
SELECT name
FROM employees
WHERE join_date > '2020-01-01';
```

6. **Write a query to list all products that have been sold more than 10 times.**
   - **Answer:**

```
SELECT product_id, COUNT(*) AS total_sales
FROM sales
GROUP BY product_id
HAVING COUNT(*) > 10;
```

7. **Write a query to get the average salary in each department.**
   - **Answer:**

```
SELECT department_id, AVG(salary) AS avg_salary
FROM employees
GROUP BY department_id;
```

8. **Write a query to find the total sales for each month.**
   - **Answer:**

```
SELECT YEAR(order_date) AS year, MONTH(order_date) AS month,
SUM(amount) AS total_sales
FROM orders
GROUP BY YEAR(order_date), MONTH(order_date);
```

9. **Write a query to get all employees who work in the "Sales" department.**
   - **Answer:**

```
SELECT * FROM employees
WHERE department = 'Sales';
```

10. **Write a query to find all customers who haven't placed any orders.**
    - **Answer:**

```
SELECT customer_id, name
```

```
FROM customers
WHERE customer_id NOT IN (SELECT customer_id FROM orders);
```