C#

1.reference type vs. value type

Variables of reference types store references to their data (objects),

Variables of value types directly contain their data.

2. boxing vs. unboxing

boxing: convert a value type to a reference type

Unboxing: convert a reference type to a value type

boxing: convert a value type to a reference type

int i = 10;

object o = i; //boxing

int j = (int) o; //unboxing

unboxing: convert the reference type to a value type

3. abstract class vs. interface

1. abstract class provides a base class to its subclasses -- use when we have a clear class hierarchy

interface defines common behaviors / functionalities that can be implemented by any classes -- work as a contract

2. one class can only inherit from one parent class, but one class can implement multiple interfaces

3. methods in abstract class can be abstract methods or non-abstract methods, but methods in interfaces are by default abstract

4. Overriding vs. Overloading

Overloading - the ability to have multiple methods within the same class with the same name, but with different parameters.

Overriding - the ability to redefine the implementation of a method in a class that inherits from a parent class.

5. What does constructor do in a class? Can it be overridden? Can it be overloaded?

constructor - used to create an object of the class and initialize class members, constructor can be overloaded, constructor cannot be inherited so a constructor cannot be overridden

6. What does static keyword do in C#?

declare a static member, which belong to the type itself rather than to a specific object

7. Difference between Virtual method and Abstract method?

abstract method - do not provide an implementation and FORCE the derived classes to override this method

virtual method - can have an implementation and provide the derived classes with the OPTION of overriding it – the child can choose to whether implement the virtual method or not

8. what are delegates in C#, what are different types of built-in delegates

Delegate: is a type safe function pointer which takes a function or a method as a parameter; is a reference type

built-in delegates:

Action: will take functions that have generics input, but return nothing (return void) as parameter

Predicate: will take functions that have generics input, and will return a Boolean value as parameter

Func: will take functions that have generics input, and will return a generics type as output to be a parameter

9. Explain different access modifiers in C#

Public – signifies that the member is accessible form outside the class’s definition and hierarchy of derived classes. A class is the encapsulation of data and the methods that works on that data

Protected – The member is not visible outside the class and can be accessed by derived classes only

Private – the member cannot be accessed outside the scope of the defining class. Therefore, not even derived classes have access to these members.

Internal – The member is visible only within the current compilation unit. The internal access modifier creates a hybrid of public and protected accessibility depending on where the code resides.

10. What is the extension method in C#? examples of built-in extension methods? How to create custom extension methods?

Extension method - a way to add new functionality into an existing type (both reference and value)

Built in extension methods - LinQ

create custom extension methods:

1. class containing extension methods must be a static class

2. method itself must be static

3. first parameter of extension method must be of the type which will be extended

4. first parameter must be written after 'this' keyword

11. Ref vs. Out vs. Params

out mode: return more than one values -- use out keyword

reference: the actual value is passed to the formal parameters so any change in formal parameters will also reflect in actual parameters -- use ref keyword

params keyword, you can specify a method parameter that takes a variable number of arguments.

12. Pass by reference vs. Pass by Value

Reference parameters - sometimes called in/out parameters because data can be transferred into the method and out again.

Value parameters - sometimes called in parameters because data can be transferred into the method but cannot be transferred out.

13. array vs. arrayList

Arrays - strongly-typed collections of the same data type and have a fixed length that cannot be changed during runtime.

An Array list - not a strongly-typed collection. It can store the values of different data types or same datatype.

14. example of encapsulation, where to implement

Private int = 1;

Setter and getter;

15. how do you handle exceptions? Syntax.

Try(){danger codes}

Catch(){Exceptions e}

Finally {will always execute whether catches exception}

16. what is generic, syntax to define

generics: allow us to design classes and methods but defer the specification or types until the class or method is declared and called

myGenerics<T>(T a, T b)

17. what is LINQ

LINQ (Language Integrated Query) is uniform query syntax in C# to save and retrieve data from different sources.

18. IEnumerable vs. IQuerable

IEnumerable - when linq is working with in-memeory data source(list, array..)

IQuerable - when linq is working with out-of-memory data source

19. First vs. FirstOrDefault vs. Single vs. SingleOrDefault

First - return the first record when there is one or more records; if nont matched --> throw an exception

FirstOrDefault - return the first record where there is one or more records; if not matched --> assign default value

Single - return the matched single record; if not matched --> throw and exception; if more than one match --> throw an exception

SingleOrDefault - return the matched single record; if not matched --> assign the default value; if more than one match --> throw an exception

20. Any vs. All

any() - check if any of the element satisfy the specific condition, if yes, return true, if no return false

all() - check if all the elements satisfy the specific condition, if yes, return true, if no, return false

21. Skip vs. Take

The Take() method extracts the first n elements (where n is a parameter to the method) from the beginning of the target sequence and returns a new sequence containing only the elements taken.

The Skip operator bypasses a specified number of contiguous rows from a sequence/table and returns the remaining table. It can skip rows from the top or can be for a certain criteria, in other words it can also skip rows depending on a certain criteria. It works like NOT IN in SQL.

22. Deferred execution and Immediate execution in LINQ

Deferred Execution

• A query variable only stores the query commands. The actual execution of the query is deferred until you iterate over the query variable in a for each statement.

• This concept is considered as deferred execution.

• Deferred execution can greatly improve performance when you have to manipulate large data collections.

• The collection results will have smaller memory foot prints.

Forcing Immediate Execution

• Queries that perform aggregation functions over a range of source elements must first iterate over those elements.

• Examples of such queries are Count, Max, Average, and First. These execute without an explicit foreach statement because the query itself must use foreach in order to return a result.

• To force immediate execution of any query and cache its results, you can call the ToList<TSource> or ToArray<TSource> methods.

A picture containing text

Description automatically generated

SQL:

1. What is index; types of indices; pros and cons

Index: On disk structure associated with a table that increase retrieval speed of rows from the tables

clustered index: physically sort the data; one table can only have one clustered index

non clustered index: will not sort the data, will be sotred separately

pro -

con - •Additional Disk Space •Insert, Update, Delete Statement become slow

2. What's the difference between Primary key and Unique constraint?

primary key vs unique key

Primary Key

•PRIMARY KEY constraints identify the column or set of columns that have values that uniquely identify a row in a table

•primary key (PK) of the table and enforces the entity integrity of the table

•A table can have only one PRIMARY KEY constraint, and a column that participates in the PRIMARY KEY constraint cannot accept null values. Because PRIMARY KEY constraints guarantee unique data, they are frequently defined on an identity column.

•If a PRIMARY KEY constraint is defined on more than one column it is called as composite primary key

•If a composite primary key is defined, values may be duplicated within one column, but each combination of values from all the columns in the PRIMARY KEY constraint definition must be unique.

•Composite primary key is a table constraint

•UNIQUE constraints enforce the uniqueness of the values in a set of columns

•In a UNIQUE constraint, no two rows in the table can have the same value for the columns

•You can use UNIQUE constraints to make sure that no duplicate values are entered in specific columns that do not participate in a primary key

•Multiple UNIQUE constraints can be defined on a table

•In a UNIQUE constraint, only one null value is allowed per column.

•A UNIQUE constraint can be referenced by a FOREIGN KEY constraint

3. Tell me about check constraint

check constraint: limit the value range that can be placed in the column

CREATE TABLE Persons (  
    ID int NOT NULL,  
    LastName varchar(255) NOT NULL,  
    FirstName varchar(255),  
    Age int,  
    CHECK (Age>=18)  
);

4. Difference between temp table and table variable

1. temp tables and tables variables are stored in temp db

2. scope: local/global; current batch

3. size: >100 rows; <100

4. usage: do not use in SP/FUNCITION; can be used in sp/function

5. Difference between WHERE and HAVING

1) both are used as filters, but having apply only to groups as a whole, and only filters on aggregation functions; where applys to individual rows

2) WHERE goes before aggregations, but HAVING filters after the aggregations

FROM/JOIN -> WHERE -> GROUP BY -> HAVING -> SELECT -> DISTINCT -> ORDER BY

|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|

cannot use alias in select

6. Difference between RANK() and DenseRank() — value gap

RANK(): if there is the same rank ,the there will be a gap for the next rank

DENSE\_RANK(): will not skip any number in rank if there is a tie

Rank() will keep the ranking, so the numbering may go 1, 2, 2, 4 etc, whereas dense\_rank will never give any gaps.

7. COUNT(\*) vs. COUNT(colName)

COUNT(\*): returns the number of rows

COUNT(colName): return the number of specified column of rows excluding NULL values

8. What's the difference between left join and inner join? JOIN and Subquery, which one has a

Better performance

JOIN: return the records that have matching values in both tables

LEFT OUTER JOIN: return all records from the left table, and the matching records from the right table,

for the non-matching records in the right table, the result set will return us null values list all customers whether they have made any purchase or not

Subquery: a SELECT statement that is embedded in a clause of another sql statement

subquery vs. join

1) JOIN can only be used in FROM; Subquery can be used in SELECT, WHERE, FROM, ORDER BY

2)subquery is easy to understand and maintain

3) usually join will have a better performance than subquery

better performance, why?

less data space need for joins

using joins, you can maximize the calculation burden on the database i.e., instead of multiple queries using one join query

SQL server have optimizer in join

9. What is correlated subquery

Correlated Subquery: inner query is dependent on the outer query

10. What is a CTE, why do we need CTE?

Common Table Expression (CTE): Specifies a temporary named result set

Improve readability and manageability of complex SQL Statements

Similar to VIEWs and even more to Derived Tables

Over time most of the CTEs will be used for this purpose

Recursive CTEs: Using CTEs and Unions, we can generate columns that can be used in the query following the CTE.

11. What does SQL Profiler do?

SQL Server Profiler: an interface to create and manage traces and analyze and replay trace results

12. What is SQL injection, how to avoid SQL injection?

SQL injection: a code injection technique used to attack data-driven applications,

in which malicious SQL statements are inserted into an entry field for execution.

Prevention:

a) stop writing dynamic queries with string concatenation;

b) prevent user supplied input which contains malicious SQL from affecting the logic of the executed query.

13. Difference between SP and user defined function? When to use SP when to use function?

sp vs. function

usage: sp for Date Manipulation Language, function for calculations

how to call: sp called by its name, function will be called in SQL query

output: sp may or maynot have output, but function must return some value

SP can function but function can not call sp

14. Criteria of Union and Union all? Difference between UNION and UNION ALL

1. both UNION and UNION ALL are used to combine multiple result sets vertically

2. criteria

the num of columns must be the same

columns types must be identical

3.alias must be given in the first SELECT statement

difference

1. UNION remove all duplicate records, but UNION ALL will not

2. UNION sort the first column ascendingly, but UNION ALL will not

3. UNION cannot be used in recursive cte, but UNION ALL can

15. Steps you take to improve SQL Queries – Rebecca

1. look at the execution plan

2. choose index wisely

3. avoid unnecessary joins

4. avoid SELECT \*

5. JOIN to replace subquery

6. derived table to avoid a lot of grouping by

16. concurrency problem in transaction

concurrency occurs when two or more transactions are trying to access the same data or info

1. dirty reads:

t1 allows t2 to read uncommitted data and then t1 rolled back

caused by isolation level read uncommitted

solved by isolation level read committed

2. lost update

t1 and t2 read and update the same data but t2 finish its work earlier than t1, then t2 will lost their update

caused by isolation level read committed

solved by isolation level repeatable read

3. non repeatable read

t1 read the same data twice while t2 is updating the data

caused by isolation level read committed

solved by isolation level repeatable read

4. phantom read

t1 reads the same data twice while t2 is inserting records

cuased by isolation level repeatable read

solved by isolation level serializable

17. what is deadlock, how to prevent

deadlock: a situation in which two or more transactions are waiting for one another to give up locks

Prevent :

Try to keep transactions short;

this will avoid holding locks in a transaction for a long period of time.

Access objects in a similar logical manner in multiple transactions.

Create a covering index to reduce the possibility of a deadlock.

18. what is normalization, 1NF - BCNF, benefits using normalization

Normalization:

•Database Normalization is a process of organizing data to minimize redundancy (data duplication), which in turn ensures data consistency.

•Normalization has a series of steps called “Forms”, the more steps you take the more normalized your tables are

19. what are the system defined databases?

A new SQL Server installation always includes four databases

❑master

❑model

❑tempdb

❑msdb

20. composite key

A composite key, in the context of relational databases, is a combination of two or more columns in a table that can be used to uniquely identify each row in the table.

Uniqueness is only guaranteed when the columns are combined; when taken individually the columns do not guarantee uniqueness.

21. candidate key

•A key that is not a PK but eligible to be a PK.

•Mostly it is a unique key without null value.

22. DDL vs. DML

Data definition language (DDL): Allows creation objects in database with:

Create, Alter, Drop

Data Control Language (DCL): allows you to determine who can see or modify the data.

GRANT, DENY, REVOKE

Data Manipulation Language (DML): Allows query and modify the data:

Select, Insert, Update, Delete.

23. ACID property – transactions

ACID

A: Atomicity -- work is atomic

C: Consistency -- whatever happends in the middle of the transaction, this property will never leave your db in half-completed state

I: Isolation -- two transactions will be isolated from each other by locking the resource

D: Durability -- once the transaction is completed, then the changes it has made to the db will be permanent

24. table scan vs. index scan

Table scan means iterate over all table rows. Index scan means iterate over all index items,

when item index meets search condition, table row is retrived through index.

Usualy index scan is less expensive than a table scan because index is more flat than a table.

25. Difference between Union and JOIN

Unions: Concatenates the results of two queries into a single result set.

JOIN: combine rows from two or more tables, based on a related column between them