* [SQL EVALUATION PREP](https://antra.talentlms.com/)

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

One of the best ways to reduce disk I/O is to use an index

Allows SQL Server to find data in a table without scanning the entire table

Example

SELECT TOP 10 p.ProductID, p.[Name], p.StandardCost

, p.[Weight], ROW\_NUMBER() OVER (ORDER BY p.Name

DESC) AS RowNumber FROM Production.Product p

ORDER BY p.Name DESC

Indexes

•Indexes are database objects based on table column for faster retrieval of data

•Query optimizer depends on indexed columns to function

•Separate structure attached to a table

•Contain pointers to the physical data

•Used to

–To quickly find data that satisfy conditions in the WHERE clause.

–To find matching rows in the JOIN clause.

–To maintain uniqueness of key column during INSERT and UPDATE.

–To Sort, Aggregate and Group data.

Types of Index

Clustered Index

–Primary Key Default (but not necessary)

–Data is stored at the leaf level

–Data is ordered by the key

Non-clustered Index

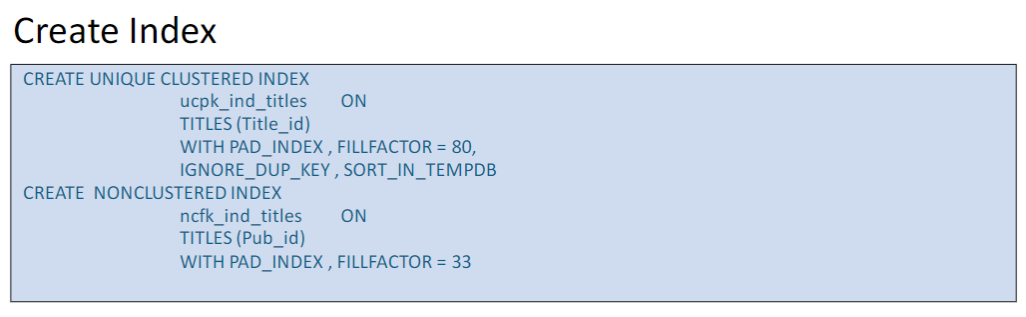
–Uses cluster key or RID of a heap

–INCLUDE stored at leaf

Disadvantages of Index

•Additional Disk Space

•Insert, Update, Delete Statement become slow



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

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

•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

--Primary key vs unique key

--1. Unique can accept one and only one null value, pk does not accept null values

--2. One table can have multiple Unique keys but only one pk

--3. Pk will sort the data by default, but unique key will not

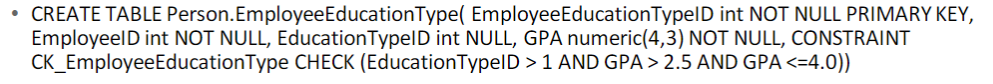
--4. PK will by default generate clustered index, but unique key will create non-clustered index

1. Tell me about check constraint

Check Constraints

•CHECK constraints enforce domain integrity by limiting the values that are accepted by a column

•They are similar to FOREIGN KEY constraints in that they control the values that are put in a column. The difference is in how they determine which values are valid: FOREIGN KEY constraints obtain the list of valid values from another table, and CHECK constraints determine the valid values from a logical expression that is not based on data in another column



1. Difference between temp table and table variable

--Temp tables vs. table variables

--1. Storage: both Temp Tables and Table Variables are stored in tempDb

--2. Scope: Temp Tables scoped local/global #, ##; Table Variables scoped for current batch

--3. Temp Tables for large data; Table Variables for smaller data

--4. Usage: Temp Tables not in Stored Procedure, Functions; Table Variables can be used

--5. Structure: Temp Tables can create index/ constraints except Foreign key; Table Variables cannot

1. Difference between WHERE and HAVING

Where: Can add filters to the variables selected. You can add more conditions with the Keywords “AND”, “Or”, and “IN” IN being used to shorten “OR” statements in situations.

Having: Can add filters to aggregate selected data

Having: Selectively Querying Grouped Data Using HAVING

The HAVING clause of the SELECT statement allows you to specify a search condition on a query using GROUP BY and/or an aggregated value

The HAVING clause is used after the GROUP BY clause. The WHERE clause, in contrast, is used to qualify the rows that are returned before the data is aggregated or grouped. HAVING qualifies the aggregated data after the data has been grouped or aggregated.

--Where and Having:

--1. Both are used as filters but having applies to groups and filters on aggreagate functions, but where applies to individual rows

--2. Where cannot filter by aggregate functions; however having can.

--3. Where can be used with Select, Update, Delete, but Having can only be used with Select

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

--Rank(): value gap

--Dense\_Rank(): No value gap

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

COUNT(\*) will count all the rows in the table, including NULL values. On the other hand, COUNT(column name) will count all the rows in the specified column while excluding NULL values.

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

inner join: return the records that have matching values in both tables

Joins will typically have better performance than subqueries

Subqueries--Select statement that is embedded in another query

--Subqueries vs Joins:

--1. Joins can only be used in the from Clause, but subqueries can be used in Select, Where, Having, From, Order By clause

--3. Joins will typically have better performance than subqueries

9. What is correlated subquery

Correlated Subquery: Subquery where inner query is dependent on the outer query

Ex:

--Correlated Subquery:

Select c.ContactName, (Select Count(o.OrderId) from Orders o where o.CustomerID = c.CustomerID) [Num of Orders]

From Customers c

Order By [Num of Orders]

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. Cte will call itself again and again until it thinks it can stop

and return something

Ex:

With OrderCte ([Total Count New], CustomerNewName)

As(

Select Count(OrderId) [Total Count], CustomerId

From Orders

Group By CustomerId)

Select c.ContactName, c.City, cte.[Total Count New]

From Customers c Left Join OrderCte cte On c.CustomerID = cte.CustomerNewName

11. What does SQL Profiler do?

<https://docs.microsoft.com/en-us/sql/tools/sql-server-profiler/sql-server-profiler?view=sql-server-ver15>

Microsoft SQL Server Profiler is a graphical user interface to SQL Trace for monitoring an instance of the Database Engine or Analysis Services. You can capture and save data about each event to a file or table to analyze later. For example, you can monitor a production environment to see which stored procedures are affecting performance by executing too slowly. SQL Server Profiler is used for activities such as:

* Stepping through problem queries to find the cause of the problem.
* Finding and diagnosing slow-running queries.
* Capturing the series of Transact-SQL statements that lead to a problem. The saved trace can then be used to replicate the problem on a test server where the problem can be diagnosed.
* Monitoring the performance of SQL Server to tune workloads. For information about tuning the physical database design for database workloads, see [Database Engine Tuning Advisor](https://docs.microsoft.com/en-us/sql/relational-databases/performance/database-engine-tuning-advisor?view=sql-server-ver15).
* Correlating performance counters to diagnose problems.

SQL Server Profiler also supports auditing the actions performed on instances of SQL Server. Audits record security-related actions for later review by a security administrator.

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

SQL injection: Some hackers inject malicious code into our SQL queries to cause harm or destruction to the database(Using Views as virtual table is easily to get injection)

We prevent SQL Injections via parameters wait till next episode of dbz, use Stored Procedures to prevent.

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

--Sp vs Function:

-- usage: sp is for DML statements while functions are used for calculations

--calling: sp uses execute, functions requires query as well as input parameters

--input: sp may or may not need any inputs , while functions do require one

--output: sp may or may not require output, but functions must have output parameters

--sp can call functions but functions cannot call SP.

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

--Union vs Union All:

--1. Union removes duplicates, union all will not

--2. Union the records from the first column is sorted automatically

--3. Unions cannot be used in recursive CTE, while Union All can

15. Steps you take to improve SQL Queries

Top 10 for Building Efficient Queries

1 Favor set-based logic over procedural or cursor logic

• The most important factor to consider when tuning queries is how to properly express logic in a set-based manner.

•Cursors or other procedural constructs limit the query optimizer’s ability to generate flexible query plans.

•Cursors can therefore reduce the possibility of performance improvements in many situations

2. Test query variations for performance

•The query optimizer can often produce widely different plans for logically equivalent queries.

•Test different techniques, such as joins or subqueries, to find out which perform betters in various situations.

3. Avoid query hints.

•You must work with the SQL Server query optimizer, rather than against it, to create efficient queries.

•Query hints tell the query optimizer how to behave and therefore override the optimizer’s ability to do its job properly.

•If you eliminate the optimizer’s choices, you might limit yourself to a query plan that is less than ideal.

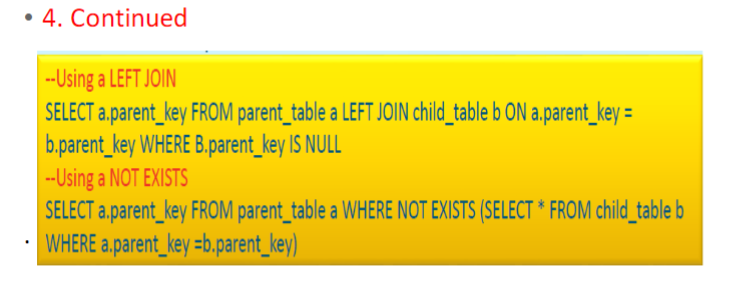
•Use query hints only when you are absolutely certain that the query optimizer is incorrect.

4. DO NOT Use correlated subqueries to improve performance.

•Since the query optimizer is able to integrate subqueries into the main query flow in a variety of ways, subqueries might help in various query tuning situations.

•Subqueries can be especially useful in situations in which you create a join to a table only to verify the

existence of correlated rows. For better performance, replace these kinds of joins with correlated subqueries that make use of the EXISTS operator

 5. Avoid using a scalar user

-defined function in the WHERE clause. Scalar user-defined functions, unlike scalar subqueries, are not optimized into the main query plan. Instead, you must call them row-by-row by using a hidden cursor.

This is especially troublesome in the WHERE clause because the function is called for every input row. Using a scalar function in the SELECT list is much less problematic because the rows have already been

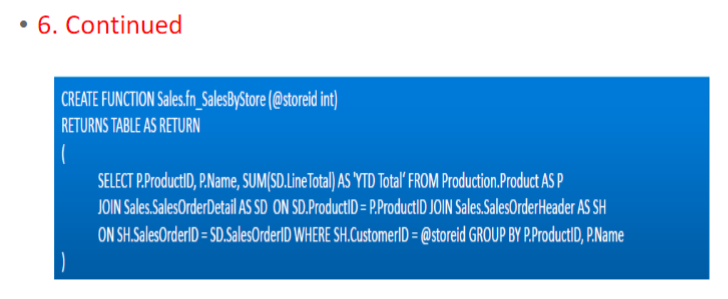
filtered in the WHERE clause

6. Use table-valued user-defined functions as derived tables.

•In contrast to scalar user-defined functions, table-valued functions are often helpful from a performance point of view when you use them as derived tables.

•The query processor evaluates a derived table only once per query.

•If you embed the logic in a table-valued user-defined function, you can encapsulate and reuse it for other queries



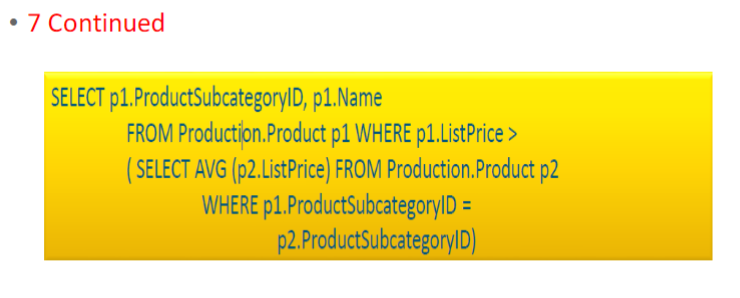
7 Avoid unnecessary GROUP BY columns

Use a subquery instead.

•The process of grouping rows becomes more expensive as you add more columns to the GROUP BY list.

•If your query has few column aggregations but many non- aggregated grouped columns, you might be able to refactor it by using a correlated scalar subquery.

•This will result in less work for grouping in the query and therefore possibly better overall query performance



8. Use CASE expressions to include variable logic in a query

The CASE expression is one of the most powerful logic tools available to T-SQL programmers.

•Using CASE, you can dynamically change column output on a row-by-row basis.

•This enables your query to return only the data that is absolutely necessary and therefore reduces the I/O operations and network overhead that is required to assemble and send large result sets to clients

9 Divide joins into temporary tables when you query very large tables.

The query optimizer’s main strategy is to find query plans that satisfy queries by using single operations.

•Although this strategy works for most cases, it can fail for larger sets of data because the huge joins require so much I/O overhead.

•In some cases, a better option is to reduce the working set by using temporary tables to materialize key parts of the query. You can then join the temporary tables to produce a final result.

10. Refactoring Cursors into Queries.

Rebuild logic as multiple queries

•Rebuild logic as a user-defined function

•Rebuild logic as a complex query with a case expression

16. concurrency problem in transaction

<https://www.geeksforgeeks.org/concurrency-problems-in-dbms-transactions/>

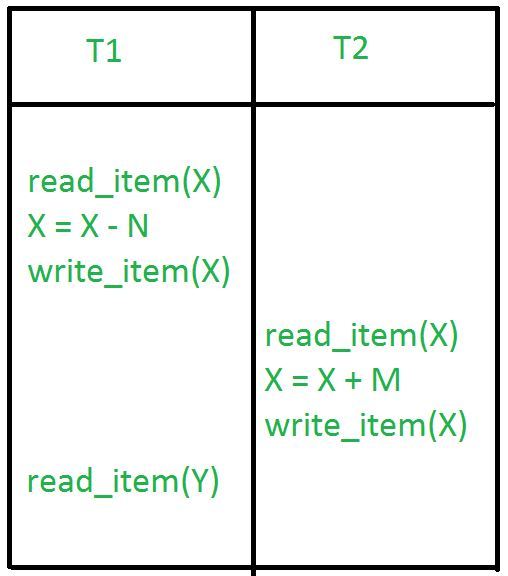
When [multiple transactions](https://www.geeksforgeeks.org/concurrency-control-in-dbms/) execute concurrently in an uncontrolled or unrestricted manner, then it might lead to several problems. These problems are commonly referred to as concurrency problems in a database environment. The five concurrency problems that can occur in the database are:

* Temporary Update Problem
* Incorrect Summary Problem
* Lost Update Problem
* Unrepeatable Read Problem
* Phantom Read Problem

### **Temporary Update Problem:**

Temporary update or dirty read problem occurs when one transaction updates an item and fails. But the updated item is used by another transaction before the item is changed or reverted back to its last value.

#### **Example:**

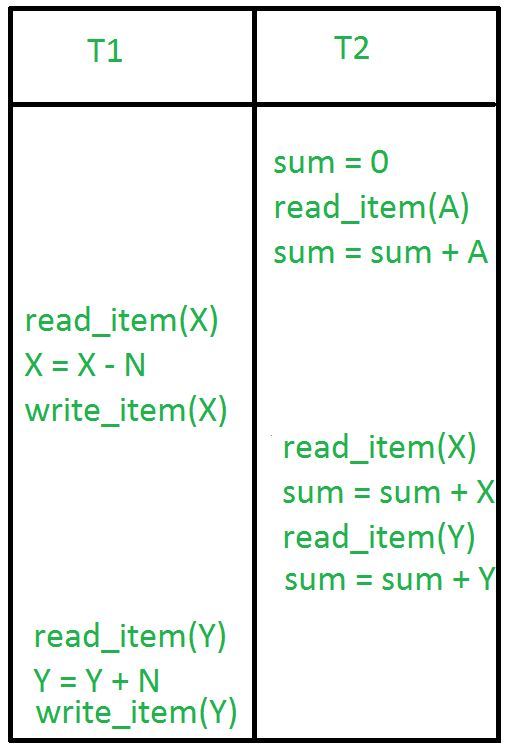


In the above example, if transaction 1 fails for some reason then X will revert back to its previous value. But transaction 2 has already read the incorrect value of X.

### **Incorrect Summary Problem:**

Consider a situation, where one transaction is applying the aggregate function on some records while another transaction is updating these records. The aggregate function may calculate some values before the values have been updated and others after they are updated.

#### **Example:**

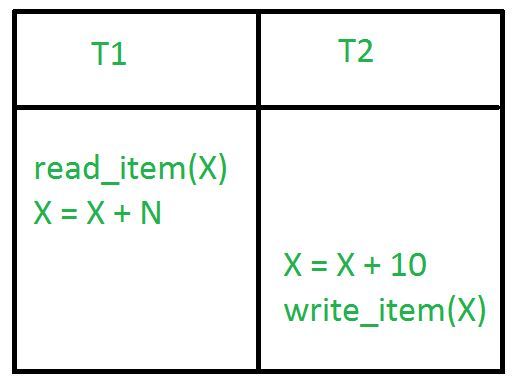


In the above example, transaction 2 is calculating the sum of some records while transaction 1 is updating them. Therefore the aggregate function may calculate some values before they have been updated and others after they have been updated.

### **Lost Update Problem:**

In the lost update problem, an update done to a data item by a transaction is lost as it is overwritten by the update done by another transaction.

#### **Example:**

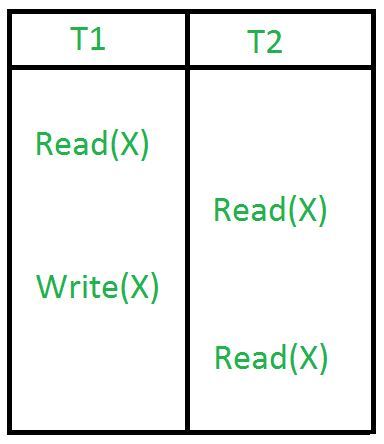


In the above example, transaction 1 changes the value of X but it gets overwritten by the update done by transaction 2 on X. Therefore, the update done by transaction 1 is lost.

### **Unrepeatable Read Problem:**

The unrepeatable problem occurs when two or more read operations of the same transaction read different values of the same variable.

#### **Example:**

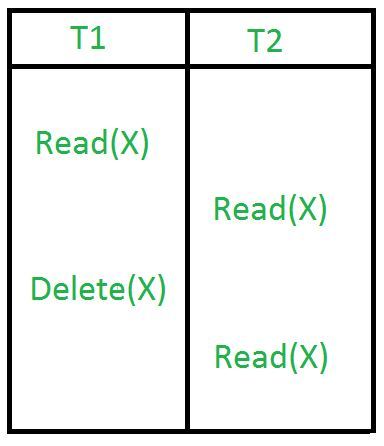


In the above example, once transaction 2 reads the variable X, a write operation in transaction 1 changes the value of the variable X. Thus, when another read operation is performed by transaction 2, it reads the new value of X which was updated by transaction 1.

### **Phantom Read Problem:**

The phantom read problem occurs when a transaction reads a variable once but when it tries to read that same variable again, an error occurs saying that the variable does not exist.

#### **Example:**



In the above example, once transaction 2 reads the variable X, transaction 1 deletes the variable X without transaction 2’s knowledge. Thus, when transaction 2 tries to read X, it is not able to do it.

17. what is deadlock, how to prevent

<https://www.geeksforgeeks.org/introduction-of-deadlock-in-operating-system/?ref=lbp>

***Deadlock***is a situation where a set of processes are blocked because each process is holding a resource and waiting for another resource acquired by some other process.

**Deadlock can arise if**the **following four conditions hold simultaneously (Necessary Conditions)**  
***Mutual Exclusion:*** Two or more resources are non-shareable (Only one process can use at a time)   
***Hold and Wait:***A process is holding at least one resource and waiting for resources.   
***No Preemption:*** A resource cannot be taken from a process unless the process releases the resource.   
***Circular Wait:*** A set of processes are waiting for each other in circular form.

**Methods for handling deadlock**   
There are three ways to handle deadlock   
1) Deadlock prevention or avoidance: The idea is to not let the system into a deadlock state.   
One can zoom into each category individually, Prevention is done by negating one of above mentioned necessary conditions for deadlock.   
Avoidance is kind of futuristic in nature. By using strategy of “Avoidance”, we have to make an assumption. We need to ensure that all information about resources which process will need are known to us prior to execution of the process. We use Banker’s algorithm (Which is in-turn a gift from Dijkstra) in order to avoid deadlock.

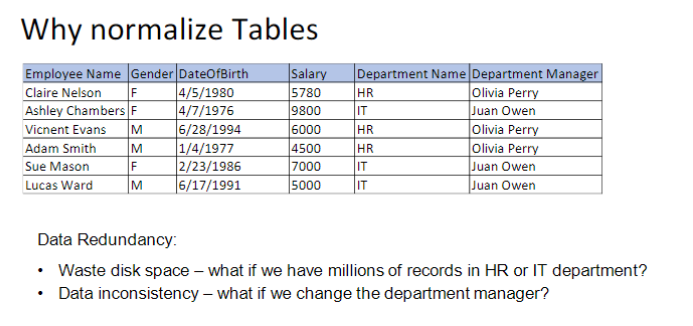
2) Deadlock detection and recovery: Let deadlock occur, then do preemption to handle it once occurred.

3) Ignore the problem altogether: If deadlock is very rare, then let it happen and reboot the system. This is the approach that both Windows and UNIX take.

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

System Databases

•A new SQL Server installation always includes four databases

❑master

❑model

❑tempdb

❑mssql system resource (hidden)

❑msdb

Master

•The master database is composed of system tables that keep track of the server installation as a whole

and all other databases that are subsequently created. Although every database has a set of system

catalogs that maintain information about objects it contains, the master database has system catalogs that

keep information about disk space, file allocations and usage, systemwide configuration settings,

endpoints, login accounts, databases on the current instance, and the existence of other SQL servers (for

distributed operations).

•The master database is critical to your system, so always keep a current backup copy of it. Operations such

as creating another database, changing configuration values, and modifying login accounts all make

modifications to master, so after performing such actions, you should back up master

Model

•The model database is simply a template database. Every time you create a new database, SQL Server makes a copy of model to form the basis of the new database. If you’d like every new database to start out with certain objects or permissions, you can put them in model, and all new databases will inherit them. You can also change most properties of the model database by using the ALTER DATABASE command, and those property values will then be used by any new database you create.

Tempdb

•The tempdb database is used as a workspace. It is unique among SQL Server databases because it’s re-created–not recovered–every time SQL Server is restarted. It’s used for temporary tables explicitly created by users, for worktables that will hold intermediate results created internally by SQL Server during query processing and sorting, for maintaining row versions used in

•The tempdb database sizing and configuration is critical for optimal functioning and performance of SQL

Server

Mssql system resource

•The mssql system resource database is a hidden database and is usually referred to as the resource database. Executable system objects, such as system stored procedures and functions, are stored here.

Msdb

•The msdb database is used by the SQL Server Agent service, which performs scheduled activities such as backups and replication tasks, and the Service Broker, which provides queuing and reliable messaging for SQL Server.

•When you are not performing backups and maintenance on this database, you should generally ignore msdb.

•All the information in msdb is accessible from Object Explorer in SQL Server Management Studio, so you usually don’t need to access the tables in this database directly. You can think of the msdb tables as another

20. composite key

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.

•This is usually seen in Joint tables.

21. candidate key

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

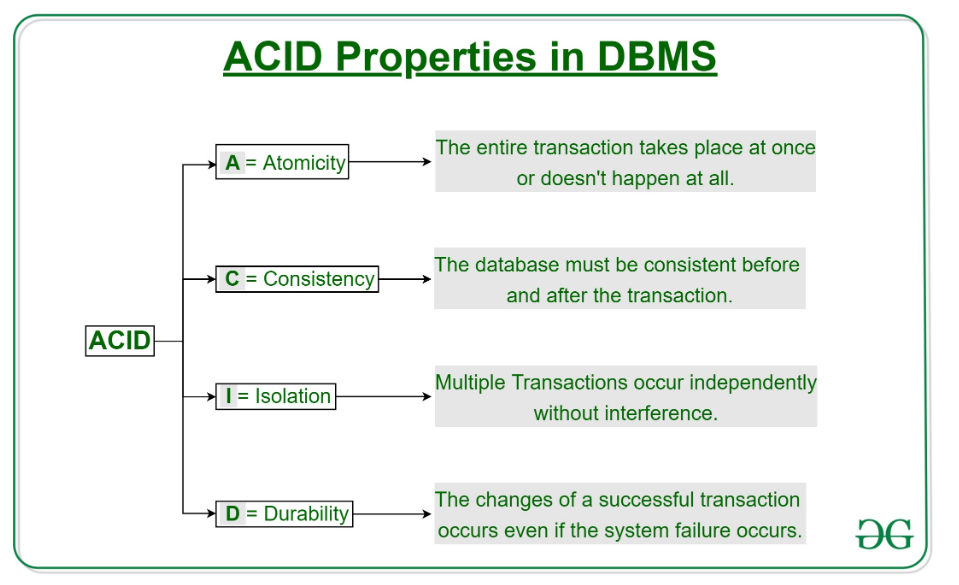
Data definition language (DDL): Allows creation objects in database with:

Create, Alter, Drop

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

            Select, Insert, Update, Delete.

23. ACID property



24. table scan vs. index scan

able scan means iterate over all table rows.

Index scan means iterate overall index items, when item index meets search condition, table row is retrieved through index.

Usually index scan is less expensive(faster) than a table scan because index is flatter than a table.

An index scan can be faster because, presumably, the index doesn't cover the entire set of columns in the table, while a table (or clustered index) scan has to read all of the data. If an index does include all of the columns in the table, then it should be roughly equivalent to a table scan, and the choice between an index scan and table (or CIX) scan will be a coin toss. The difference is that when you have fewer columns in the index, you can fit more index rows on an 8kb page, leading to fewer overall pages you have to read in order to scan all of the data in the index.

25. Difference between Union and JOIN

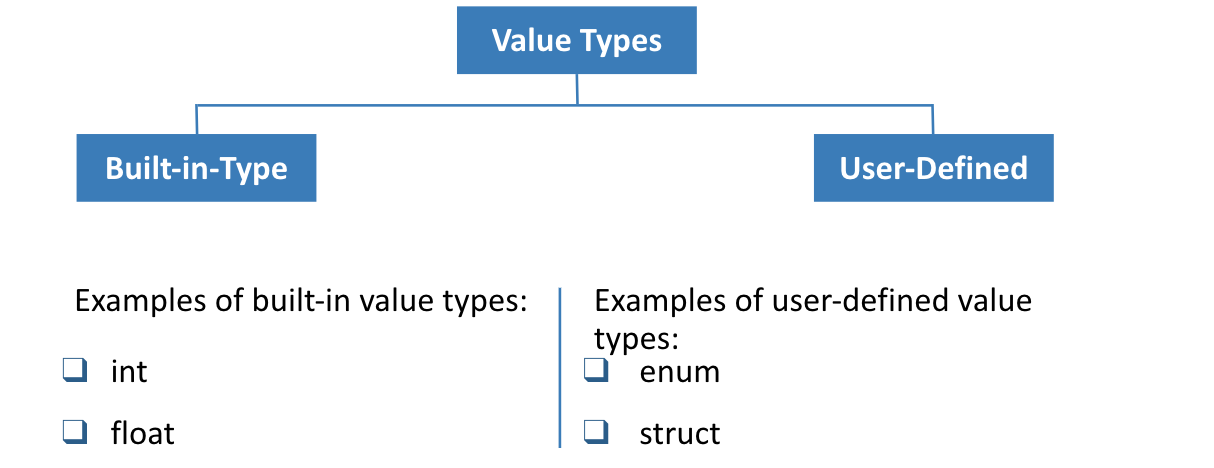
joins combine data into new columns.

Unions combine data into new rows.

**C# evaluation prep**

* 1. reference type vs. value type

|  |  |
| --- | --- |
| Value Types | Reference Types |
| Directly contain their data | Store references to their data (Known as objects) |
| Each has its own copy of data | Two reference variable can reference the same object |
| Operation on one cannot effect another | Operation on one can effect another |



* 1. boxing vs. unboxing

int num = 10;

Object obj = num; // Boxing

int i = (int)obj; //Unboxing

- So Avoid Boxing and unboxing when not necessary, use generics collections.

- Make the value type simple.

- Or you have to use object. unless you have to cast it,

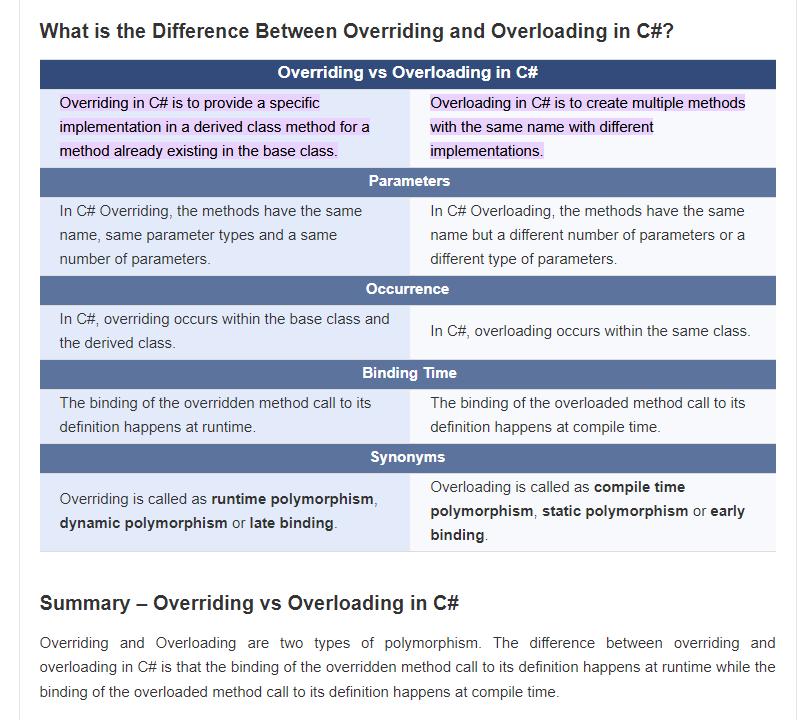
Source: https://stackoverflow.com/questions/1028520/use-cases-for-boxing-a-value-type-in-c

3. abstract class vs. interface

## Interface vs Abstract Class

* Interface supports multiple inheritance but abstract class does not
* Interface cannot have instanced constructor but abstract class can have
* Interface has by default all members abstract and public, but abstract class can have abstract and concrete members
* Interface cannot have fields but abstract class can have fields.

4. Overriding vs. Overloading



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

Constructor:

It is a special method which shares the same name as the class and doesn’t have a return type, not even void

Constructors is used to create an object of the class and initialize class members

if there is no constructor in the class, the compiler provides a default constructor

Default constructors does not take any input parameters

If we create any other constructor, the default constructor will be replaced

Constructors can be overloaded

Constructors cannot be inherited so a constructor cannot be overridden

By default, derived class constructor will make a call to the base class parameter less constructor

--Static constructor will run first before other constructors

Polymorphism:

Compile Time- Method Overloading:

Runtime Time- Method Overriding:

1. What does static keyword do in C#?

<https://www.geeksforgeeks.org/static-keyword-in-c-sharp/>

<https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/classes-and-structs/static-classes-and-static-class-members>

 a static class cannot be instantiated. In other words, you cannot use the [new](https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/operators/new-operator) operator to create a variable of the class type. Because there is no instance variable, you access the members of a static class by using the class name itself. For example, if you have a static class that is named UtilityClass that has a public static method named MethodA, you call the method as shown in the following example:

UtilityClass.MethodA();

Static, in C#, is a keyword that can be used to declare a member of a type so that it is specific to that type. The static modifier can be used with a class, field, method, property, operator, event or constructor.

•Static constructor: initializes static members of the type; parameter less

 The advantage of using a static class is that the compiler can check to make sure that no instance members are accidentally added. The compiler will guarantee that instances of this class cannot be created.

Static classes are sealed and therefore cannot be inherited. They cannot inherit from any class except [Object](https://docs.microsoft.com/en-us/dotnet/api/system.object).

1. Difference between Virtual method and Abstract method?

**Abstract method**

Abstract method is a method which can not have body.

It must be implemented in the derived class.

It is by default virtual so we can not use virtual keyword with an abstract method

If a method is marked abstract then the class containing it must be marked as abstract

•Abstract method

•Only be contained in an abstract class.

•No implementation

•Must be overridden in derived classes

•Virtual Method

•Can be contained in both abstract class and concrete classes

•Has default implementation

•Overriding in derived classes is optional

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

# Delegate(communication/callback between 2 parties)

* A delegate is a type safe function pointer.
* The signature of delegate must match the signature of function, the delegate points to.
* Delegates are pointers to function. Delegates are used for implementing events and call back methods.

# Anonymous, Lamda, Action, Prediction and Func are different types of built-in delegates

**Func Delegate**

Func is generic delegate present in System namespace. It takes one or more input parameters and returns one out parameter. The last parameter is considered as a return value.

Func delegate type can include 0 to 16 input parameters of different types. It must have one return type. So return type is mandatory but input parameter is not.

**Action Delegate**  
Action is a generic delegate present in System namespace. It takes one or more input parameters and returns nothing. So it does not return any value.

**predicate Delegat**  
Predicate delegate is also inbuilt generic delegate and is present in System namespace.  
It is used to verify certain criteria of method and returns output as Boolean, either True or False. Predicate can be used with method, anonymous and lambda expression.  
**Example 1:** Check String value is number using Predicate delegates.  
Predicate predicate = DelegateClass.IsNumeric;  
bool number = predicate("1234")

1. Explain different access modifiers in C#

•Access Modifiers:

•Public: accessible everywhere

•Protected: accessible in the same class or derived class

•Internal: accessible in the same assembly

•Private: accessible only in current class

1. What is the extension method in C#? examples of built-in extension methods?

Extension method is a way to add new functionality into an existing type (both reference and value). LINQ and Dapper are based on Extension methods. We have used extension methods to create custom html helpers in MVC. To create an extension method follow these four steps

Class containing extension method must be static.

Extension method must be static

First parameter to the extension method must be the type to be extended

First parameter must be written after this keyword

LINQ is a bunch of extension methods that provide addtion functionalities on IEnumerable<T>

1. How to create custom extension methods?

<https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/classes-and-structs/how-to-implement-and-call-a-custom-extension-method>

**Extension method**

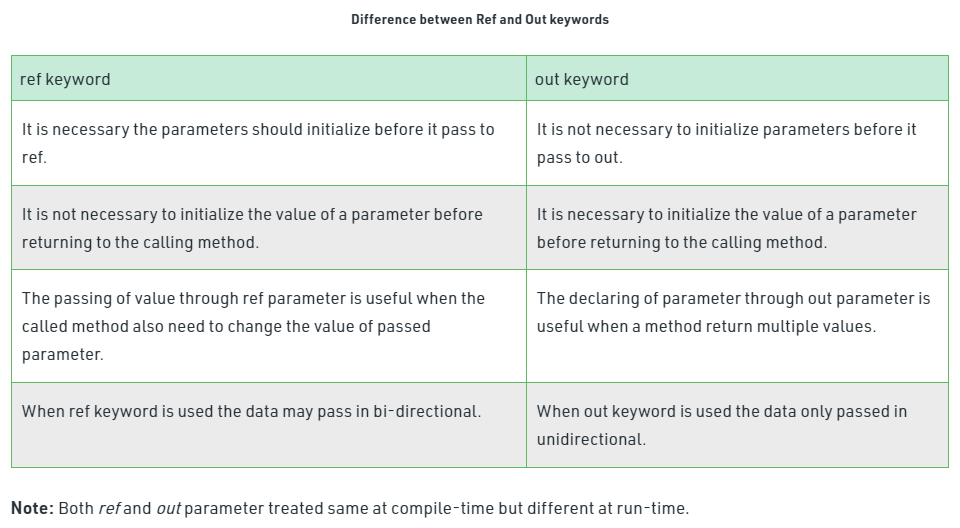
Extension method is a way to add new functionality into an existing type (both reference and value). LINQ and Dapper are based on Extension methods. We have used extension methods to create custom html helpers in MVC. To create an extension method follow these four steps

* Class containing extension method must be static.
* Extension method must be static
* First parameter to the extension method must be the type to be extended
* First parameter must be written after this keyword

1. Ref vs. Out vs. Params

<https://www.geeksforgeeks.org/difference-between-ref-and-out-keywords-in-c-sharp/>

The **out**is a keyword in C# which is used for the passing the arguments to methods as a reference type. It is generally used when a method returns multiple values. The out parameter does not pass the property.

The **ref**is a keyword in C# which is used for the passing the arguments by a reference. Or we can say that if any changes made in this argument in the method will reflect in that variable when the control return to the calling method. The ref parameter does not pass the [property](https://www.geeksforgeeks.org/c-properties/). 

1. Pass by reference vs. Pass by Value

## Passing Parameters

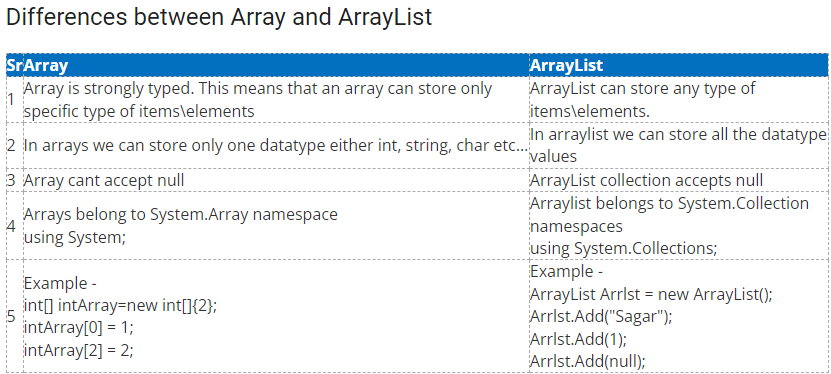
* By value

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

* By reference

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

1. array vs. arrayList



1. example of encapsulation, where to implement

Encapsulation: hide data implementation

When we creating Entity, wrapping up of data under a signle unit. Data hiding. Increased Flexibility Reusability, Testing code is easier.

•Get & set accessor

•Access Modifiers:

•Public: accessible everywhere

•Protected: accessible in the same class or derived class

•Internal: accessible in the same assembly

•Private: accessible only in current class

// C# program to illustrate the

// read and write property

using System;

public class Student {

// Declare name field

private string name = "Solution";

// Declare name property

public string Name

{

get

{

return name;

}

set

{

name = value;

}

}

}

class TestStudent {

// Main Method

public static void Main(string[] args)

{

Student s = new Student();

// calls set accessor of the property Name,

// and pass "Solution" as value of the

// standard field 'value'.

s.Name = "Solution";

// displays Solution, Calls the get accessor

// of the property Name.

Console.WriteLine("Name: " + s.Name);

}

}

1. how do you handle exceptions? Syntax.

The Try...Catch...Finally control structure is fundamental to structured exception handling

It tests a piece of code, filters exceptions created by the execution of that code, and reacts differently based on the type of thrown exception.

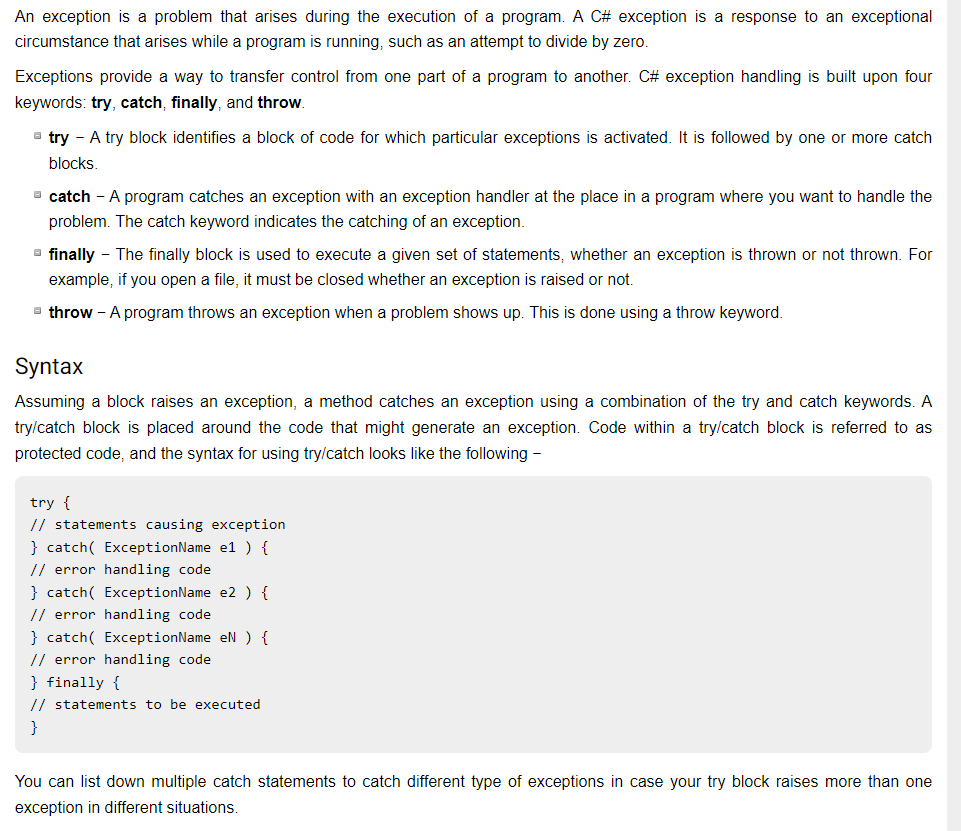
The Try statement provides the code that is being tested for exceptions

Catch clauses identify blocks of code that are associated with specific exceptions series of Catch blocks

The Finally statement contains code that executes regardless of whether or not an exception occurs within the Try block

We can have nested try blocks

Catch Blocks will in order of most specific to most generic

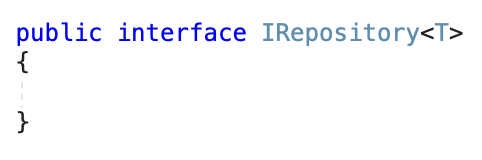


1. what is generic, syntax to define

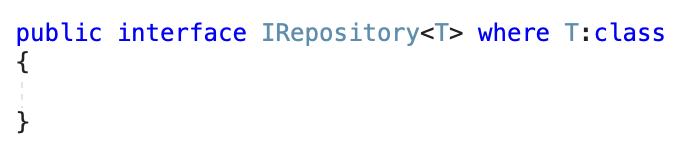
<https://www.geeksforgeeks.org/c-sharp-generics-introduction/#:~:text=Generic%20is%20a%20class%20which,methods%2C%20classes%2C%20and%20interfaces>.

Generic allows you to use a class or method that can work with any data type. You need to define the data type while using class or method. Usually the type is specified by the <T>.  You can make a class, interface or method generic.

Generic types perform better than normal system types because they reduce the need for boxing, unboxing, and type casting the variables or objects.



You can also restrict a generic to accept value type or reference type using where



once a type is generic you can use the data type with the methods inside that

