International Journal of Allied Practice, Research and Review
Website: www.ijaprr.com (ISSN 2350-1294)

Building Database Performances Using Managed Code

Junaida shafi[1], Iqra Jan[2]
[1]CSE Department, Krukshetra University, Krukshetra , Haryana
[2]CSE Department, Krukshetra University, Krukshetra, Haryana

Abstract - The database plays a very important role in modern application development. Working on databases requires deep knowledge of database programming. To make database programming more efficient and versatile we need to integrate general programing concepts in database programming and the CLR integration is the best example by which managed code can be correlated with database programming to achieve high performance and flexibility.

Keywords: Assembly;CLR Integration; optimization; database programming; managed code.

I. Introduction

Database performance is the biggest challenge for organizations, especially Database administrators and managers. Developers and Database operating professionals both agree that achieving and maintaining high database performance is their biggest challenge related to persistence. Most of the times surveys have been done where performance is the biggest challenge related to the database. The database becomes easily accessible when we talk of programming in database. Programming plays a massive role in database management system. It makes the toughest jobs very easy for us by making use of different and efficient programming concepts like variables, conditional constructs, looping constructs, exceptions with procedures, functions etc. The errors and delays result in slow database performance.To get the better results we have to wait for several seconds in the best case.In the worst case scenario, we usually find the wrong data or no results at all. Smart development needs efficient query writing and so far my research is concerned I am working on a concept that enables database developers specifically using MS SQL Server 2012 or above to use minimum queries for maximum work. In simple words we can say a Query optimizer for database programming. The best way to tune performance is to write queries in a number of ways and compare their reads and execution plans .In the modern era, digital data is considered as the more valuable asset of an organization, and the organizations assign more significance to it than the software and hardware assets. Database systems are computer-based record keeping systems, which have been developed to store data for efficient retrieval and processing. Optimization primarily means selection, followed by sequencing in specific order, of the different SQL clauses to formulate an efficient query from the multiple query plans by drawing a comparison of the query plans based on the cost of the resources involved and the response time. The objective of optimization is to provide minimum response time and maximum throughput (i.e., the efficient use of resources). The SQL developers and database administrators (DBAs) used an important skill in to increase the performance and they found query optimization as the best tool. And for database developers and DBAs must understand the query optimization in order to prepare a best query execution plan. CLR (Common Language Runtime) is one of the most mandatory components of .NET framework that affords an environment for the application to run. It provides us the various featureslike exception handling, security, debugging .the most

important features delivered by it are memory management; type system, language interoperability, platform independence and security management.CLR can multitude a selection of languages. It provides a common set of tools across these languages which ensure us the interoperability of codes between the languages. The code that is being developed that targets CLR with a language compiler is called a managed code.CLR is such an environment that accomplishes the codes written in any .NET programming language.it allows database designers to create objects in any of the .NET supported languages (e.g., C# etc.) and inserts the objects in the database .Such a database object is called a managed database object. To implement programming logics we use triggers, procedures, user-defined functions in SQL Server 2012.however, it is not possible to implement the required functionality by using T-SQL code in some situations We can insert these programs in our database so that they can run in the same environment in which the database exists. Microsoft.Sql.Server.Server namespace includes core functionality for CLR programming

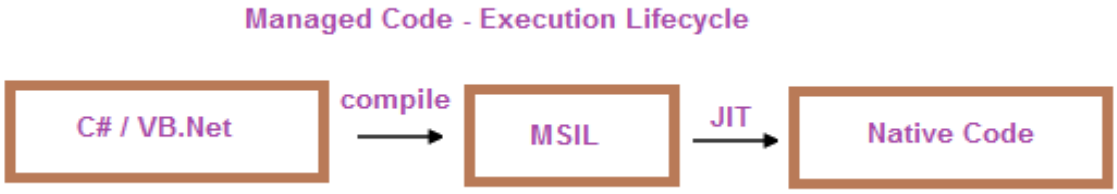
The database becomes easily accessible when we talk of programming in database. Programming plays a massive role in database management system. It makes the toughest jobs very easy for us by making use of different and efficient programming concepts like variables, conditional constructs, looping constructs, exceptions with procedures, functions etc. Slow database performance could mean errors and delays in accessing information. In best case scenario, itmeans we have to wait several seconds for a database search to return the correct result.

There are two main components of DBMS to evaluate and optimize the query: the query optimizer and the query execution engine. The execution engine uses physical operators like SORT, NESTED LOOP JOIN, MERGE JOIN and INDEX SCAN, etc. to take input and produce the required output. These physical operators are the building blocks which make the SQL query execution possible. These operators construct a tree called parsed tree, which represents the flow of data from one operator to the others in the form of edges moving back and forth to the nodes. Query optimizer takes a parsed tree of the SQL query as an input from the execution engine and produces an best possible or close to optimal execution plan out of the possible execution plans for the given query based on the least resource consumption. For a given query, there are many logical algebraic representations and there are many choices of physical operators to implement these logical representations in addition to the variation of response time of these plans. Therefore, it is obviously not an easy task for an optimizer to generate an optimal plan.

Query optimizer being an important module of a DBMS has a greater impact on the database performance. It analyzes a number of candidate plans generated for a given query which have equivalent output but having different resource costs. It selects an efficient plan out of these candidates' plans having the least cost. Although much work has been performed in query optimization, but cardinality estimation to manage optimization time and effective cost estimation with respect to the server state are still the challenging problems. The optimizer should be capable to handle complex and large data by adopting some specific search strategy to counter the problems that an optimizer may face. Genetic strategies can be used to solve the optimization issues such as joins which is a challenging part for an optimizer in making efficient plan.

II. The concept of Managed code

Managed code is the code that is used to aim the facilities of the managed runtime execution environment such as Common Language Runtime in .Net Technology. The Managed Code running under a Common Language Runtime cannot be edited outside the runtime environment as well as cannot call straight from outside the runtime environment. It refers to a bond of cooperation between natively executing code and the runtime. It offers services like garbage collection, run-time type checking, reference checking etc. Managed code can avoid many usual programming mistakes that leads to uneven applicationssuch as type safety checking, memory management, destruction of unused Objects etc.



Managed Code is the code thatC# compilers create. It assembles to Intermediate Language (IL), not to machine code that could run directly on your computer. The CIL describes the classes, methods, and attributes of the code that are being created is kept in a file known as Assembly along with the metadata.

When the assembly runs, the runtime stays to offer services such as security, memory management, threading, and the like. The application is managed by the runtime.C# can yield only managed code. When we create a project, select one of the application types whose name starts with .Managed. such as .Managed C++ application.

To implement CLR integration in SQL Server 2012, we have to follow the following steps:

Step 1:first create a managed code in any of the .NET programming languages for creating database objects in SQL Server. Managed code holds classes and methods that offer a desired functionality.

Step 2: After creating a managed code now we have to create an assembly. Assemblies are not executed directly in SQL Server. Therefore we need to import and configure the assemblies in the database engine before using the assemblies. Whenever we import an assembly, its details are added to the sys.assemblies.system table. We can import a .NET assembly in SQL Server database engine using CREATE ASSEMBLY statement.

Step 3: After importing assemblies, we can create managed database objects that use the managed code provided in the assembly. By default, SQL Server does not allow the running of managed code on the server. So, we need to enable the CLR integration feature in our database before creating a managed database object.

While developing managed database objects, we will have to use the System.Data.SqlClient,System.Data.SqlTypes, and Microsoft.SqlServer.Server namespaces found in .NET base class libraries. Depending on the requirements, the database developer can create the following types of objects:

1. Triggers.
2. User Defined Functions.
3. User-Defined Types.
4. Stored Procedures

A trigger is a special type of stored procedure that automatically fires when a language event executes. Managed triggers are used for fulfilling advanced trigger logic that cannot be done using TSQL. For creating managed triggers we have to follow the following steps:

(a) Create a .NET class that implements the functionality of trigger. Then compile that class to produce a .NET assembly.

(b) Register that assembly in SQL Server using CREATE ASSEMBLY statement.

(C) Create a trigger and associate it with the actual methods of the assembly

User Defined Functions show a significant role in SQL Server. User Defined functions can accept parameters and return data. For most of the situations we have to write complex logic which cannot be written using a single query. In such situations, UDFs play a keyrole. User defined functions in SQL Server is of two types:

1. Scalar functions: those functions which return a single value.

2. Table Valued Functions: that function which returns a row set of SQL server Table data type.

There are various steps to create the managed code:

(a) Create a .NET class that implements the functionality of the user-defined function. Then, compile that class to produce a .NET assembly.

(b) Register that assembly in SQL Server using CREATE ASSEMBLY statement.

(c) Create user-defined function and associate it with the actual methods of the assembly.

UDTs can contain multiple elements differentiating them from the traditional alias data types which consist of a single SQL Server system data type. Because UDTs are retrieved by the system as a whole, their use for complex data types may negatively influence performance. Complex data is generally best showed using traditional rows and tables. UDTs in SQL Server are well matched to the following:

- Date, time, currency, and extended numeric types
- Geospatial applications
- Encoded or encrypted data

The process of developing UDTs in SQL Server consists of the following steps:

1. **Code and build the assembly that defines the UDT.** UDTs are defined using any of the languages supported by the.NET Framework common language runtime (CLR) that produce supportable code. This includes Visual C# and Visual Basic .NET. The data is visible as fields and properties of a .NET Framework class or structure, and behaviors are defined by methods of the class or structure.

2. **Register the assembly.** UDTs can be organized through the Visual Studio user interface in a database project, or by using the Transact-SQL CREATE ASSEMBLY statement, which copies the assembly containing the class or structure into a database.

3. **Create the UDT in SQL Server.** Once an assembly is loaded into a host database, we use the Transact-SQL CREATE TYPE statement to create a UDT and expose the members of the class or structure as members of the UDT. UDTs exist only in the frameworkof a single database, and, once registered, have no dependencies on the external files from which they were created.

   We can create a data type definition in any of the .NET supported languages and use it as a data type within SQL Server. This data type can be a mixture of any other existing data type with more alterations applied on it. We have to perform the following steps to create user-defined data type:

   (a) Create a .NET class that implements the functionality of the user-defined type. Then, compile that class to produce a .NET assembly.

   (b) Register that assembly in SQL Server using the CREATE ASSEMBLY statement. (c) Create a user-defined data type that refers to the registered assembly

A stored procedure is a pre-complied object and is a group of TSQL statements compiled into a single execution plan. Stored procedures are used for security a reason that is to prevent the user from hacking.

Microsoft SQL Server stored procedures return data in four ways:

- Output parameters, which can return either data (such as an integer or character value) or a cursor variable (cursors are result sets that can be retrieved one row at a time).
- Return codes, which are always an integer value.
- A result set for each SELECT statement contained in the stored procedure or any other stored procedures called by the stored procedure.
- A global cursor that can be referenced outside the stored procedure.

Stored procedures can also guard users from requiringknowing the details of the tables in the database. If a set of stored procedures supports all of the business functions users need to achieve, users never need to access the tables directly; they can just execute the stored procedures that model the business processes with which they are familiar.

we can use managed code to be executed as a stored procedure with the CLR integration. For this purpose, we have to create a procedure that refers to an imported assembly. To create a stored procedure using the managed code, we have to perform the following steps:

(a) Create a .NET class that implements the functionality of the stored procedure. Then, compile that class to produce a .NET assembly.

(b) Register that assembly in SQL Server using the CREATE ASSEMBLY statement.

(c) Create a stored procedure and associate the stored procedure with the actual methods of the assembly

III. Advantages of using Managed code

TSQL is specifically intended for direct data access and manipulation in the database. While Transact-SQL excels at data access and management, it is not a complete programming language. For example, Transact-SQL does not provision arrays, collections, for-each loops, bit shifting, or classes. While some of these constructs can be replicated in Transact-SQL, managed code has integrated support for these constructs. Depending on the situation, these features can provide a captivating reason to implement certain database functionality in managed code. .NET supported languages offer object-oriented capabilities such as encapsulation, inheritance, and polymorphism. Using these capabilities related code can now be easily organized into classes and namespaces. When we are working with large amounts of server code, this allows us to more easily organize and maintain our code. Managed code is better suited than Transact-SQL for calculations and complicated execution logic, and features extensive support for many complex tasks, including string handling and regular expressions. With the functionality found in the .NET Framework Library, we have access to thousands of pre-built classes and routines. These can be easily accessed from any stored procedure, trigger or user defined function. The Base Class Library (BCL) includes classes that provide functionality for string manipulation, advanced math operations, file access, and more. One of the profits of managed code is type safety, or the assurance that code accesses types only in well-defined, permissible ways. Before managed code is executed, the CLR validates that the code is safe. For example, the code is tested to ensure that no memory is read that has not previously been written. The CLR can also help ensure that code does not operate unmanaged memory.CLR integration offers the promising for improved performance. Another important element in our resolution about whether to use TSQL or managed code is where we would

like our code to reside, the server computer or the client computer. Both TSQL and managed code can be route on the server. This places code and data close together, and allows us to take advantage of the processing power of the server Most client computers today are very dominant, and we may wish to take advantage of this processing power by placing as much code as possible on the client. Managed code can run on a client computer, while TSQL cannot.

IV. CONCLUSION

Database performance is very important aspect in organization which can be increased in number of ways .query optimization is one of the possible ways of increasing the performance in databases. Different ways of queries are used to increase the optimization in databases. Even concept of managed code can also be used to optimize the performance. Common Language Runtime (CLR) is the heart of .net framework which serves as the main gateway between c# and the database (MSSQL Server 2012) through which optimization can also be achieved directly without making the dynamic linking library(DLL) file in c#.

V. ACKNOWLEDGMENT

I wish to acknowledge all the people who have worked in Database Technology and optimization whose work inspired me to write this research paper inbuilding database performances using managed code. I also would like to acknowledge my parents, my colleagues, my teachers and especially my better half Mr.Usaid who supported me to write my 2nd research paper and put my ideas and research on paper as a contribution to coming generation.

VI. REFERENCES

[1]Leonard G.Lobel, Andrew J.Brust "Programming SQL Server 2012".

[2] Grant Fritchey "SQL Server Execution Plans".

[3]Ross Mistry, Stacia Misner "Introducing MS SQL SERVER 2012".

[4] Patrick Leblane" MS SQL server 2012 step by step".

[5]Paul Atkinson, Robert Vieira" Beginning MS SQL Server 2102 Programming".

[6] Gregajerkic, MatijaLah, DejonSarka" Implementing a Data Warehouse with SQL Server 2012".

[7]Brian knights, Devin Knight, Davis Wayne "KNIGH'S MICROSOFT SQL SERVER 2012 INTEGRATION SERVICE 24_HOUR TRAINER".

[8] ItzikBen_Gan"Microsoft SQL Server 2012 High Performance T-Sql".

[9] William R Stanek"Microsoft SQL Server 2012 Pocket Consultant".

[10] Robert Vieira "Beginning Microsoft SQL Server 2008 Programming".

[11] Erik Veerman "Microsoft SQL Server 2008 Integration Services: Problem, Design, Solution".

[12] Brian Knight "Knight's 24-Hour Trainer: Microsoft SQL Server 2008 Integration Services".

[13] Paul Nielsen "Microsoft SQL Server 2008 Bible".

[14] Robert Vieira "Professional Microsoft SQL Server 2008 Programming".

[15] Paul Turley "Beginning T-SQL with Microsoft SQL Server 2005 and 2008".

[16] Erik Veerman "Professional Microsoft SQL Server 2008 Integration Services".

[17] Christian Nagel, Bill Enjen, Jay Glynn, Morgan Skinner, Karli Watson “Professional C# 2008”, Wiley Publications

[18] MichealOtey “Microsoft SQL server 2005 New features”.

[19]. M. Jarke and J. Koch,” Query Optimization in Database Systems”, Computing Surveys, Vol. 16, No. 2, June 1964.

[20] Jonathan Goldstein and Per-Åke Larson (2001), “optimizing queries”, ACM SIGMOD 2001 May 21-24, Santa Barbara, California, USA©2001 1-58113-332-4/01/05.

[21]. Alvin Cheung, Armando Solar-Lezama, Samuel Madden (2013),”optimizing database applications”, PLDI’13, June 16–19, 2013, Seattle, WA, USA.Copyright 2013 ACM 978-1-4503-2014-6/13/06.

[22].Navita kumara (2012),”SQL server query optimization techniques”, International Journal of Scientific and Research Publications, Volume 2, Issue 6, June 2012 2 ISSN 2250-3153.

[23]. Costel Gabriel CORLĂŢAN, Marius Mihai LAZĂR, Valentina LUCA, Octavian Teodor PETRICICĂ (),”Query Optimization Techniques in Microsoft SQL Server”,Database Systems Journal vol. V, no. 2/2014.