**CHAPTER TWO**

**LITERATURE REVIEW**

**2.0 PREAMBLE**

**2.1 WHAT IS LITERATURE REVIEW**

Literature review is an account of what has been published on a topic by accredited scholars and researchers. It is also the text of scholar which includes the current knowledge including substantive findings, as well as theoretical and methodological contribution to a particular topic.

According to Renom, J. (2001). Literature review is the effective evaluation of selected documents on a research topic. A review may form an essentials part of the research process or may constitute a research project in itself.

The purpose of writing this literature review is to pass on to the reader what knowledge and ideas have been established on the topic and what their strengths weakness and literature review lets the researcher gain and demonstrates skills like the ability to skills like the ability to seek information and to scan the literature efficiently using manual or computerized method and to identify useful articles and books.

**2.2 REVIEW OF RELATED LITERATURE**

Electronic Commerce (e-commerce) applications support the interaction between different parties participating in a commerce transaction via the network, as well as the management of the data involved in the process [2]. The increasing importance of e-commerce is apparent in the study conducted by researchers at the GVU (Graphics, Visualization, and Usability) Center at the Georgia Institute of Technology. In their summary of the findings from the eighth survey, the researchers report that "e-commerce is taking off both in terms of the number of users  
2 shopping as well as the total amount people are spending via Internet based transactions". Over three quarters of the 10,000 respondents report having purchased items online. The most cited reason for using the web for personal shopping was convenience (65%), followed by availability of vendor information (60%), no pressure from sales person (55%) and saving time (53%). Although the issue of security remains the primary reasons why more people do not purchase items online, the GVA survey also indicates that faith in the security of ecommerce is increasing. As more people gain confidence in current encryption technologies, more and more users can be expected to frequently purchase items online.

A good e-commerce site should present the following factors to the customers for better usability:

* Knowing when an item was saved or not saved in the shopping cart.
* Returning to different parts of the site after adding an item to the shopping cart.
* Easy scanning and selecting items in a list.
* Effective categorical organization of products.
* Simple navigation from home page to information and order links for specific products.
* Obvious shopping links or buttons.
* Minimal and effective security notifications or messages.
* Consistent layout of product information.

Another important factor in the design of an e-commerce site is feedback. The interactive cycle between a user and a web site is not complete until the web site responds to a command entered by the user. According to Norman, "feedback-sending back to the user information about what action has actually been done, what result has been accomplished--is a well known concept in the science of control and information theory. Imagine trying to talk to someone when you cannot even hear your own voice, or trying to draw a picture with a pencil that leaves no mark: there would be no feedback". Web site feedback often consists of a change in the visual or verbal information presented to the user. Simple examples include highlighting a selection made by the user or filling a field on a form based on a user's selection from a pull down list. Another example is using the sound of a cash register to confirm that a product has been added to an electronic shopping cart. Completed orders should be acknowledged quickly. This may be done with an acknowledgment or fulfillment page. The amount of time it takes to generate and download this page, however, is a source of irritation for many e-commerce users. Users are quick to attribute meaning to events. A blank page, or what a user perceives to be "a long time" to receive an acknowledgment, may be interpreted as "there must be something wrong with the order." If generating an acknowledgment may take longer than what may be reasonably expected by the user, then the design should include intermediate feedback to the user indicating the progress being made toward acknowledgment or fulfillment. Finally, feedback should not distract the user. Actions and reactions made by the web site should be meaningful. Feedback should not draw the user's attention away from the important tasks of gathering information, selecting products, and placing orders.

Hewlett M. (1993) says: Bookshop entails buying and selling of books of all sorts with the buyer still making his choice and making the payment through the internet or through any other computer network. This pattern of trade conducted through the internet has grown extraordinarily simultaneously with wide spread of the internet. This new trend has contributed immensely to the growth of E-business and particularly in E-education. The number of transaction grows every day, spurring and drawing innovations in the E-library system and E-commerce, Electronic fund transfer, supply chain management, On-line transaction processing and internet marketing.

This chapter of the project research does not only explore the background of the study, the statement of the problems, purpose of study, significance of the study and the limitation, it also go as far as exposing and expounding the secrets and fortunes locked up in an online book shopping.

**2.3 REASON FOR ONLINE BOOKSHOP**

The reason for online bookshop can be basically grouped either as fundamental or secondary. The fundamental reasons are aimed at forestalling the possibility of getting right amount of book, magazine and journals at exact need. On the other hand the secondary reason his aimed at ensuring favorable return on investments.

The desire of an enterprise to hold book can be delivered into the following:

1. **Maximizing Space**

As a small business owner, you probably have limited storage space in your business location. According to store, online bookshop allows you to maximize your space by identifying the faster and slower sellers in your book mix. As a result, you can provide for space for better seller while weeding out the slow moving items.

1. **Room for new Merchandise**

In a competitive business environment, being the first to carry the newest books on the market gives you an edge over your competitors, by effectively managing and controlling your books, journal and magazine.

1. **Turning over book**

Score also indicate that property online bookshop ensures increased speed in turning over your materials. These reduce the cost associated moving through your operation instead of collecting dust in your bookshop.

1. **Discounting Merchandise**

According to small business town, Perhaps an item only sell on a seasonal basis, or no longer meets the needs of your costumer reduce price, by removing the item from yourself or bookshop and displaying it at a reduced price,

1. **Avoiding lost sale**

These means just carry much material can lead to higher expenses for your business, nor carry enough of the right materials can also pose a problem if you don’t control your materials, you can run the risk of missing out on sales due to running out of a key item. If consumers can’t find what they need, they will probably go somewhere else. They may decide not to patronize your business in the future, so this cause cost of sales.

**2.4 DATABASES**

In the early days of computerization, it was normal to maintain specific files for individual application. Data were processes centrally in batches and there was little or no online interrogation of data. This approach is wholly inefficient for most today’s data processing systems. Supporting this, version (1991) enumerated the problems that result from organizing data using the file systems.

1. There exists a high redundancy between files, which result from that fact that the information is replicated in different places, and that these replications are not controlled by a central monitor.
2. Inconsistencies might result to possibilities that a program makes change on the files it uses without these changes being made at the same time by all other programs that usues the file.
3. There exists inflexibility against changes in the application if new actions or events arise in the course of time, these can be realized at a substantial expense of time.
4. The work of many programmers involved is characterized by low productivity since program maintenance is expensive if the structure of an existing file has to be modified during its lifetime, then all application programs has to be modified correspondingly.
5. Finally, there is the problem of adopting and maintaining standards with respect to coding, data formats e.t.c, which is important for exchanging data or for migration to a new operating system release, or even to a new computer system.

To overcome these problems, databases were developed. It is now common for large organization to organize their operational data using the database technology.

The subject of database is adequately covered in many works on database technology. Clifton (1983) briefly defines database as a collection of data supporting the operation of an organization. Quoting CIMA, Lucey (1991) provides a more detailed definition:

A database is a file of data structured in such a way that it may serve a number of applications without its structure being dedicated by any one of those applications, the concepts being that programs are written round the database rather than the file being structure to meet the needs of particular programs.

Russell, M. (1987) dealt extensively on the needs for use of computers on such database system like computerized clearance system. In the words of Dimorji (2003). “At the center of any information system is a database, which is any collection of related information grouped together as a simple item. The term can also apply to the ways in which information is catalogued, analyzed, stored and use manually”.

Russell (2005) was also of the view that without computer, effective handling of candidates’ records cannot be achieved effectively.

In a database, all the data is define together rather than each file being defined separately; in fact, all the literature consulted seems to support the fact that a database is a collection of structure data with the structure of data independent of any particular application. Specifying the need for database, O’Leary (1996) listed the following advantages:

1. **Sharing:** In an organization, information in one department can be readily share with others
2. **Security:** Users are given password or access only to the kind of information they need to know. Thus, the payroll department may have access to employees’ pay rate, but other department would not.
3. **Fewer file:** With several department having access to one file, there are fewer files. Therefore, excess storage or what is called ‘redundancy’ is reduced.
4. **Data integrity:** Older file system many times did not have integrity. That is, a change made in the file of other department might not be made in the file in another department. As one might expect, this can cause serious problems and conflicts when data is use for important decisions affecting both departments.

To the advantages enumerated above, Vossen (1991) adds:

1. Standards/access protocols can be enforced.
2. Currency of data can be maintained.
3. Data/program independence can be maintained
4. Conflicting requirements can be balanced among users.

In these days of integrated networks, the database appears has the most logical method for organizing the operational data of large organizations. One may as well say that these advantages give the database the attractions over the traditional file processing method.

**2.5 INTEGRATING THE WEBSITE AND DATABASE**

Customers ordering from an e-commerce website need to be able to get information about a vendor’s products and services, ask questions, select items they wish to purchase, and submit payment information. Vendors need to be able to track customer inquiries and preferences and process their orders. So a well organized database is essential for the development and maintenance of an e-commerce site. In a static Web page, content is determined at the time when the page is created. As users access a static page, the page always displays the same information. Example of a static Web page is the page displaying company information. In a dynamic Web page, content varies based on user input and data received from external sources. We use the term “data-based Web pages” to refer to dynamic Web pages deriving some or all of their content from data files or databases. A data-based Web page is requested when a user clicks a hyperlink or the submit button on a Web page form. If the request comes from clicking a hyperlink, the link specifies either a Web server program or a Web page that calls a Web server program. In some cases, the program performs a static query, such as “Display all items from the Inventory”. Although this query requires no user input, the results vary depending on when the query is made. If the request is generated when the user clicks a form’s submit button, instead of a hyperlink, the Web server program typically uses the form inputs to create a query. For example, the user might select five books to be purchased and then submit the input to the Web server program. The Web server program then services the order, generating a dynamic Web page response to confirm the transaction. In either case, the Web server is responsible for formatting the query results by adding HTML tags. The Web server program then sends the program’s output back to the client’s browser as a Web page.

**2.6 THEORETICAL TECHNOLOGY FOR DEVELOPMENT**

**2.6.1 C#, Programming Language**

C# [(/si: ʃɑːrp/)](https://en.wikipedia.org/wiki/Help:IPA/English) is a [multi-paradigm programming language](https://en.wikipedia.org/wiki/Multi-paradigm_programming_language) encompassing [strong typing](https://en.wikipedia.org/wiki/Strong_typing), [imperative](https://en.wikipedia.org/wiki/Imperative_programming), [declarative](https://en.wikipedia.org/wiki/Declarative_programming), [functional](https://en.wikipedia.org/wiki/Functional_programming), [generic](https://en.wikipedia.org/wiki/Generic_programming), [object-oriented](https://en.wikipedia.org/wiki/Object-oriented_programming) ([class](https://en.wikipedia.org/wiki/Class_(computer_science))-based), and [component-oriented](https://en.wikipedia.org/wiki/Component-based_software_engineering) programming disciplines. It was developed by [Microsoft](https://en.wikipedia.org/wiki/Microsoft) within its [.NET](https://en.wikipedia.org/wiki/.NET_Framework) initiative and later approved as a standard by [Ecma](https://en.wikipedia.org/wiki/Ecma_International" \o "Ecma International) (ECMA-334) and [ISO](https://en.wikipedia.org/wiki/International_Organization_for_Standardization) (ISO/IEC 23270:2006). C# is one of the programming languages designed for the [Common Language Infrastructure](https://en.wikipedia.org/wiki/Common_Language_Infrastructure).

C# is a general-purpose, object-oriented programming language. Its development team is led by [Anders Hejlsberg](https://en.wikipedia.org/wiki/Anders_Hejlsberg). The most recent version is C# 7.2, which was released in 2017 along with [Visual Studio](https://en.wikipedia.org/wiki/Microsoft_Visual_Studio) 2017 version 15.5.

**2.6.1.1 DESIGN GOAL**

The ECMA standard lists these design goals for C#:

* The language is intended to be a simple, modern, general-purpose, [object-oriented programming](https://en.wikipedia.org/wiki/Object-oriented_programming) language.
* The language, and implementations thereof, should provide support for software engineering principles such as [strong type](https://en.wikipedia.org/wiki/Strong_type)checking, array [bounds checking](https://en.wikipedia.org/wiki/Bounds_checking), detection of attempts to use [uninitialized variables](https://en.wikipedia.org/wiki/Uninitialized_variable), and automatic [garbage collection](https://en.wikipedia.org/wiki/Garbage_collection_(computer_science)). Software robustness, durability, and programmer productivity are important.
* The language is intended for use in developing [software components](https://en.wikipedia.org/wiki/Software_components) suitable for deployment in distributed environments.
* Portability is very important for source code and programmers, especially those already familiar with [C](https://en.wikipedia.org/wiki/C_(programming_language)) and [C++](https://en.wikipedia.org/wiki/C%2B%2B).
* Support for [internationalization](https://en.wikipedia.org/wiki/Internationalization_and_localization) is very important.
* C# is intended to be suitable for writing applications for both hosted and [embedded systems](https://en.wikipedia.org/wiki/Embedded_system), ranging from the very large that use sophisticated [operating systems](https://en.wikipedia.org/wiki/Operating_system), down to the very small having dedicated functions.

**2.6.1.2 HISTORY**

During the development of the .NET Framework, the [class libraries](https://en.wikipedia.org/wiki/Base_Class_Library) were originally written using a [managed code](https://en.wikipedia.org/wiki/Managed_code) compiler system called *Simple Managed C* (SMC). In January 1999, [Anders Hejlsberg](https://en.wikipedia.org/wiki/Anders_Hejlsberg) formed a team to build a new language at the time called Cool, which stood for "[C-like](https://en.wikipedia.org/wiki/C-like) Object Oriented Language".[[19]](https://en.wikipedia.org/wiki/C_Sharp_(programming_language)#cite_note-computerworld2008-21) Microsoft had considered keeping the name "Cool" as the final name of the language, but chose not to do so for trademark reasons. By the time the .NET project was publicly announced at the July 2000 [Professional Developers Conference](https://en.wikipedia.org/wiki/Professional_Developers_Conference), the language had been renamed C#, and the class libraries and [ASP.NET](https://en.wikipedia.org/wiki/ASP.NET) runtime had been ported to C#.

Hejlsberg is C#'s principal designer and lead architect at Microsoft, and was previously involved with the design of [Turbo Pascal](https://en.wikipedia.org/wiki/Turbo_Pascal), [Embarcadero Delphi](https://en.wikipedia.org/wiki/Embarcadero_Delphi) (formerly CodeGear Delphi, Inprise Delphi and Borland Delphi), and [Visual J++](https://en.wikipedia.org/wiki/Visual_J%2B%2B). In interviews and technical papers he has stated that flaws[[*citation needed*](https://en.wikipedia.org/wiki/Wikipedia:Citation_needed)] in most major programming languages (e.g. [C++](https://en.wikipedia.org/wiki/C%2B%2B), [Java](https://en.wikipedia.org/wiki/Java_(programming_language)), [Delphi](https://en.wikipedia.org/wiki/Embarcadero_Delphi), and [Smalltalk](https://en.wikipedia.org/wiki/Smalltalk)) drove the fundamentals of the [Common Language Runtime](https://en.wikipedia.org/wiki/Common_Language_Runtime) (CLR), which, in turn, drove the design of the C# language itself.

Since the release of C# 2.0 in November 2005, the C# and Java languages have evolved on increasingly divergent trajectories, becoming two very different languages. One of the first major departures came with the addition of [generics](https://en.wikipedia.org/wiki/Generic_programming) to both languages, with vastly different implementations. C# makes use of [reification](https://en.wikipedia.org/wiki/Reification_(computer_science)) to provide "first-class" generic objects that can be used like any other class, with code generation performed at class-load time.[[25]](https://en.wikipedia.org/wiki/C_Sharp_(programming_language)#cite_note-27) Furthermore, C# has added several major features to accommodate functional-style programming, culminating in the [LINQ](https://en.wikipedia.org/wiki/Language_Integrated_Query) extensions released with C# 3.0 and its supporting framework of [lambda expressions](https://en.wikipedia.org/wiki/Lambda_expressions), [extension methods](https://en.wikipedia.org/wiki/Extension_method), and [anonymous types](https://en.wikipedia.org/wiki/Anonymous_type).

These features enable C# programmers to use functional programming techniques, such as [closures](https://en.wikipedia.org/wiki/Closure_(computer_science)), when it is advantageous to their application. The LINQ extensions and the functional imports help developers reduce the amount of [boilerplate code](https://en.wikipedia.org/wiki/Boilerplate_code) that is included in common tasks like querying a database, parsing an xml file, or searching through a data structure, shifting the emphasis onto the actual program logic to help improve readability and maintainability.

**2.6.1.3 NAME**

The name "C sharp" was inspired by musical notation where a [sharp](https://en.wikipedia.org/wiki/Sharp_(music)) indicates that the written note should be made a [semitone](https://en.wikipedia.org/wiki/Semitone) higher in [pitch](https://en.wikipedia.org/wiki/Pitch_(music)). This is similar to the language name of [C++](https://en.wikipedia.org/wiki/C%2B%2B), where "++" indicates that a variable should be incremented by 1. The sharp symbol also resembles a [ligature](https://en.wikipedia.org/wiki/Typographic_ligature) of four "+" symbols (in a two-by-two grid), further implying that the language is an increment of C++.

**2.6.1.4 SYNTAX**

The core syntax of C# language is similar to that of other C-style languages such as C, C++ and Java. In particular:

* Semicolons are used to denote the end of a statement.
* [Curly brackets](https://en.wikipedia.org/wiki/Curly_brackets) are used to group statements. Statements are commonly grouped into methods (functions), methods into classes, and classes into [namespaces](https://en.wikipedia.org/wiki/Namespaces).
* Variables are assigned using an [equals sign](https://en.wikipedia.org/wiki/Equals_sign), but compared using [two consecutive equals signs](https://en.wikipedia.org/wiki/%3D%3D).
* [Square brackets](https://en.wikipedia.org/wiki/Square_brackets) are used with [arrays](https://en.wikipedia.org/wiki/Array_data_structure), both to declare them and to get a value at a given index in one of them.

**2.6.1.5 EXAMPLES**

The following is a very simple C# program, a version of the classic "[Hello world](https://en.wikipedia.org/wiki/Hello_world)" example:

using System;

class Program

{

static void Main(string[] args)

{

Console.WriteLine("Hello, world!");

}

}

What will display on the program is:

Hello, world!

Each line has a purpose:

**Using System;**

The above line of code tells the compiler to use System as a candidate prefix for types used in the source code. In this case, when the compiler sees use of the Console type later in the source code, it tries to find a type named Console, first in the current assembly, followed by all referenced assemblies. In this case the compiler fails to find such a type, since the name of the type is actually System.Console. The compiler then attempts to find a type named System.Console by using the System prefix from the using statement, and this time it succeeds. The using statement allows the programmer to state all candidate prefixes to use during compilation instead of always using full type names.

**Class Program**

Above is a [class](https://en.wikipedia.org/wiki/Class_(computer_science)) definition. Everything between the following pair of braces describes Program.

**static void main(string[] args)**

This declares the class member method where the program begins execution. The .NET runtime calls the Main method. (Note: Main may also be called from elsewhere, like any other method, e.g. from another method of Program.) The [static keyword](https://en.wikipedia.org/wiki/Method_(computer_programming)#Static_methods) makes the method accessible without an instance of Program. Each console application's Main entry point must be declared static. Otherwise, the program would require an instance, but any instance would require a program. To avoid that irresolvable [circular dependency](https://en.wikipedia.org/wiki/Circular_dependency), C# compilers processing [console applications](https://en.wikipedia.org/wiki/Console_application) (like that above) report an error, if there is no static Main method. The void keyword declares that Main has no [return value](https://en.wikipedia.org/wiki/Return_value).

**Console.WriteLine("Hello, world!");**

This line writes the output. Console is a static class in the System namespace. It provides an interface to the standard input, output, and error streams for console applications. The program calls the Console method WriteLine, which displays on the console a line with the argument, the string "Hello, world!".

**2.6.1.6 C# Version**

**C# Version 1**

C# version 1 really did look an awful lot like Java.  As [part of its stated design goals for ECMA](http://feeldotneteasy.blogspot.com/2011/01/c-design-goals.html), it sought to be a “simple, modern, general purpose object-oriented language.”  At the time, it could have done worse thank looking like Java in order to achieve those goals.

But if you looked back on C# 1.0 now, you’d find yourself a little dizzy.  It lacked the built in async capabilities and some of the slick functionality around generics that we take for granted.  As a matter of fact, it lacked generics altogether.  And [Linq](https://msdn.microsoft.com/en-us/library/bb308959.aspx)?  Nope.  That would take some years to come out.

**C# Version 2**

Now things start to get interesting.  Let’s take a look at some major features of C# 2.0, released in 2005, along with Visual Studio 2005.  (Check out [the book by NDepend creator Patrick Smacchia](http://www.ndepend.com/practicalbook) about .NET 2.0.)

* [Generics](https://www.tutorialspoint.com/csharp/csharp_generics.htm)
* [Partial types](https://www.dotnetperls.com/partial)
* [Anonymous methods](https://www.tutorialspoint.com/csharp/csharp_anonymous_methods.htm)
* [Nullable types](https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/nullable-types/)
* [Iterators](https://msdn.microsoft.com/en-us/library/65zzykke(v=vs.100).aspx)
* [Covariance and contravariance](https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/concepts/covariance-contravariance/)

While Microsoft may have started with a pretty generic object-oriented language, C# Version 2 changed that in a hurry.  Once they had their feet under them, they went after some serious developer pain points.  And they went after them in a big way.

With generics, you have types and methods that can operate on an arbitrary type while still retaining type safety.  So, for instance, having a List<T> lets you have List<string> or List<int>  and perform type safe operations on those strings or ints while you iterate through them.  This certainly beats creating ListInt inheritors or casting from Object for every operation.

**C# Version 3**

C# Version 3 came in late 2007, along with Visual Studio 2008, though the full boat of language features would actually come with C# Version 3.5.  And what a version this proved to be.  I would go so far as to say that this established C# as a truly formidable programming language.  Let’s take a look at some major features in this version.

* [Auto implemented properties](https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/classes-and-structs/auto-implemented-properties)
* [Anonymous types](http://www.c-sharpcorner.com/UploadFile/ff2f08/anonymous-types-in-C-Sharp/)
* [Query expressions](https://docs.microsoft.com/en-us/dotnet/csharp/linq/query-expression-basics)
* [Lambda expression](http://www.daedtech.com/introduction-to-c-lambda-expressions/)
* [Expression trees](https://blogs.msdn.microsoft.com/charlie/2008/01/31/expression-tree-basics/)
* [Extension methods](https://www.codeproject.com/Tips/709310/Extension-Method-In-Csharp)

**C# Version 4**

C# Version 4 would have had a difficult time living up to the groundbreaking status of version 3.  With version 3, Microsoft had moved the language firmly out from the shadow of Java and into prominence.  The language was quickly becoming elegant.

The next version did introduce some cool stuff, though.

* [Dynamic binding](https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/keywords/dynamic)
* [Named/optional arguments](https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/classes-and-structs/named-and-optional-arguments)
* [Generic covariant and contravariant](https://docs.microsoft.com/en-us/dotnet/standard/generics/covariance-and-contravariance)
* [Embedded interop types](https://stackoverflow.com/questions/20514240/whats-the-difference-setting-embed-interop-types-true-and-false-in-visual-studi)

**C# Version 5**

With C# Version 5, Microsoft released a very focused version of the language.  They put nearly all of their effort for that version into another pretty groundbreaking language concept.  Here is the major features list.

* [Asynchronous members](https://msdn.microsoft.com/library/hh191443(vs.110).aspx)

**C# Version 6**

With versions 3 and 5, Microsoft had done some pretty impressive stuff in an OO language.  (Version 2 did as well, but they were fast following Java with those language features.)  With version 6, they would go away from doing a dominant killer feature and instead release a lot of features that delighted users of the language.  Here are some of them.

* [Static imports (a la Java)](http://geekswithblogs.net/BlackRabbitCoder/archive/2015/04/16/c.net-little-wonders-static-using-statements-in-c-6.aspx)
* [Exception filters](https://www.thomaslevesque.com/2015/06/21/exception-filters-in-c-6/)
* [Property initializers](http://geekswithblogs.net/WinAZ/archive/2015/06/30/whatrsquos-new-in-c-6.0-auto-property-initializers.aspx)
* [Expression bodied members](https://lostechies.com/jimmybogard/2015/12/17/c-6-feature-review-expression-bodied-function-members/)
* [Null propagator](https://davefancher.com/2014/08/14/c-6-0-null-propagation-operator/)
* [String interpolation](https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/keywords/interpolated-strings)
* [nameof operator](https://stackoverflow.com/questions/31695900/what-is-the-purpose-of-nameof)
* [Dictionary initializer](https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/classes-and-structs/how-to-initialize-a-dictionary-with-a-collection-initializer)

**C# Version 7**

Finally, we arrive at C# version 7.  That’s the current version as of the writing of this post.  This has some evolutionary and cool stuff in the vein of C# 6, but without the compiler as a service.  Here are some of the new features.

* [Out variables](http://www.c-sharpcorner.com/article/out-variables-in-c-sharp-7-0/)
* [Tuples and deconstruction](https://www.thomaslevesque.com/2016/08/23/tuple-deconstruction-in-c-7/)
* [Pattern matching](https://docs.microsoft.com/en-us/dotnet/csharp/whats-new/csharp-7#pattern-matching)
* [Local functions](http://www.infoworld.com/article/3182416/application-development/c-7-in-depth-exploring-local-functions.html)
* [Expanded expression bodied members](https://docs.microsoft.com/en-us/dotnet/csharp/whats-new/csharp-7#more-expression-bodied-members)
* [Ref locals and returns](https://docs.microsoft.com/en-us/dotnet/csharp/whats-new/csharp-7#ref-locals-and-returns)

**2.6.2 TRANSACT-SQL** (**T-SQL**)

It is [Microsoft](https://en.wikipedia.org/wiki/Microsoft)'s and [Sybase](https://en.wikipedia.org/wiki/Sybase)'s proprietary extension to the [SQL](https://en.wikipedia.org/wiki/SQL) (Structured Query Language) used to interact with [relational databases](https://en.wikipedia.org/wiki/Relational_database). T-SQL expands on the SQL standard to include [procedural](https://en.wikipedia.org/wiki/Procedural_programming) programming, [local variables](https://en.wikipedia.org/wiki/Local_variable), various support functions for string processing, date processing, mathematics, etc. and changes to the [DELETE](https://en.wikipedia.org/wiki/Delete_(SQL)) and [UPDATE](https://en.wikipedia.org/wiki/Update_(SQL)) statements.

Transact-SQL is central to using [Microsoft SQL Server](https://en.wikipedia.org/wiki/Microsoft_SQL_Server). All applications that communicate with an instance of SQL Server do so by sending Transact-SQL statements to the server, regardless of the user interface of the application. [Stored procedures](https://en.wikipedia.org/wiki/Stored_procedure) in SQL Server are executable server-side routines. The advantage of stored procedures is the ability to pass parameters.

**2.6.3 MICROSOFT SQL SERVER**

Microsoft SQL Server is a [relational database management system](https://en.wikipedia.org/wiki/Relational_database_management_system) developed by [Microsoft](https://en.wikipedia.org/wiki/Microsoft). As a [database server](https://en.wikipedia.org/wiki/Database_server), it is a [software product](https://en.wikipedia.org/wiki/Software_product) with the primary function of storing and retrieving data as requested by other [software applications](https://en.wikipedia.org/wiki/Software_application)—which may run either on the same computer or on another computer across a network (including the Internet).

Microsoft markets at least a dozen different editions of Microsoft SQL Server, aimed at different audiences and for workloads ranging from small single-machine applications to large Internet-facing applications with many [concurrent users](https://en.wikipedia.org/wiki/Concurrent_user).

The history of Microsoft SQL Server begins with the first Microsoft SQL Server product – SQL Server 1.0, a 16-bit server for the [OS/2](https://en.wikipedia.org/wiki/OS/2) operating system in 1989 - and extends to the current day.

Milestones

* MS SQL Server for OS/2 began as a project to port [Sybase SQL Server](https://en.wikipedia.org/wiki/Sybase_SQL_Server) onto OS/2 in 1989, by [Sybase](https://en.wikipedia.org/wiki/Sybase), [Ashton-Tate](https://en.wikipedia.org/wiki/Ashton-Tate), and [Microsoft](https://en.wikipedia.org/wiki/Microsoft).
* SQL Server 4.2 for NT is released in 1993, marking the entry onto [Windows NT](https://en.wikipedia.org/wiki/Windows_NT).
* SQL Server 6.0 is released in 1995, marking the end of collaboration with [Sybase](https://en.wikipedia.org/wiki/Sybase); Sybase would continue developing their own variant of *SQL Server*, Sybase [Adaptive Server Enterprise](https://en.wikipedia.org/wiki/Adaptive_Server_Enterprise), independently of Microsoft.
* SQL Server 7.0 is released in 1998, marking the conversion of the source code from C to C++.
* SQL Server 2005, released in 2005, finishes the complete revision of the old Sybase code into Microsoft code.
* SQL Server 2017, released in 2017, adds Linux support for these Linux platforms: [Red Hat Enterprise Linux](https://en.wikipedia.org/wiki/Red_Hat_Enterprise_Linux), [SUSE Linux Enterprise Server](https://en.wikipedia.org/wiki/SUSE_Linux_Enterprise_Server), [Ubuntu](https://en.wikipedia.org/wiki/Ubuntu_(operating_system)) & [Docker Engine](https://en.wikipedia.org/wiki/Docker_(software)" \o "Docker (software)).[[3]](https://en.wikipedia.org/wiki/Microsoft_SQL_Server#cite_note-3)

Currently

As of October 2017 the following versions are supported by Microsoft:

* SQL Server 2008
* SQL Server 2008 R2
* SQL Server 2012
* SQL Server 2014
* SQL Server 2016
* SQL Server 2017