

|  |  |  |
| --- | --- | --- |
|  |  | |
|  |  |  | | |  | |
|  |  | Computer Research Project  The difficulties in learning SQL, as a newbie to programming or an experienced developer, and reasons. | | |  | |
|  |  | | | | |  |
|  | Author: K.P.I. Shenesh Perera  Date: 02/06/2019  IDM | | | | |  |
|  |  | | |  | | |

Contents

[Research Problem 2](#_Toc15227097)

[Literature Review 3](#_Toc15227098)

[Structured Query Language - The declarative language that runs databases. 3](#_Toc15227099)

[Choice of DBMS for discussion purposes: Microsoft SQL Server 4](#_Toc15227100)

[Data Corruption, a problem every developer faces 5](#_Toc15227101)

[SQL Error 5172 5](#_Toc15227102)

[SQL Fatal 823 Error 5](#_Toc15227103)

[Stackoverflow – where developers learn, share and build careers 6](#_Toc15227104)

[Object Oriented Programming – Imperative programming, an example of the opposite of declarative programming 6](#_Toc15227105)

[Primary & Secondary Research Data Collection Methodologies 8](#_Toc15227106)

[Sampling existing system documentation 8](#_Toc15227107)

[Research through visiting sites 8](#_Toc15227108)

[Observation of the work environment where the problem exists 8](#_Toc15227109)

[Questionnaires 9](#_Toc15227110)

[Interviews 9](#_Toc15227111)

[Prototyping 10](#_Toc15227112)

[Merits, pitfalls and limitations in above research data collection methods through compare & contrast 10](#_Toc15227113)

[Prototyping vs Sampling existing documentation 10](#_Toc15227114)

[Research through site visiting vs observation of work environment 11](#_Toc15227115)

[Questionnaires vs Interviews 11](#_Toc15227116)

[Primary Research – Questionnaire 12](#_Toc15227117)

# 

# Research Problem

The research will be conducted in order to identify “**The difficulties in learning SQL, as a newbie to programming or an experienced developer, and reasons**”. The reason as to why such a research problem was chosen by my hand was because I quite often saw in the Diploma classes and even HND classes of where I learn at (IDM Negombo), quite often struggle at learning or even understanding certain SQL concepts. **As such my audience is obviously developers.** It wasn’t just SQL, it was the whole context of database programming, however though since I do not have sufficient time to conduct a research of that magnitude, I slimmed down and focused on the main culprit which Is SQL. So the research question is “**Is SQL difficult to learn?**”.

However, the main objectives of this research are to find out:

* Why it is difficult to learn SQL?
* If SQL is demanded in the job industry?
* Are there pitfalls developers (of various skill levels) fall into while learning SQL?

# Literature Review

The literature review section of this document is composed information that is considered supplementary in order to understand the research outcomes and the conclusions I take from the outcomes that I will describe.

As such the literature review of this documentation has the following goals:

* Provide background/basic information that the reader of this research documentation requires in order to understand research outcomes
* Provide context to the description of the research documentation, such as to make the reader find more sense in the purpose of this research
* To ensure that the reader is confident and can rely on this research documentation for their own goals and purposes.

## Structured Query Language - The declarative language that runs databases.

Pronounced Sequel or “es-kue-el” is the language by which machines converse with databases. According to the American National Standards Institute, SQL is the standard for interactions with Relational Database Management Systems. One of the most basic yet powerful features of SQL is to allowing developers to perform Create, Read, Update and Delete (CRUD) actions on a database.

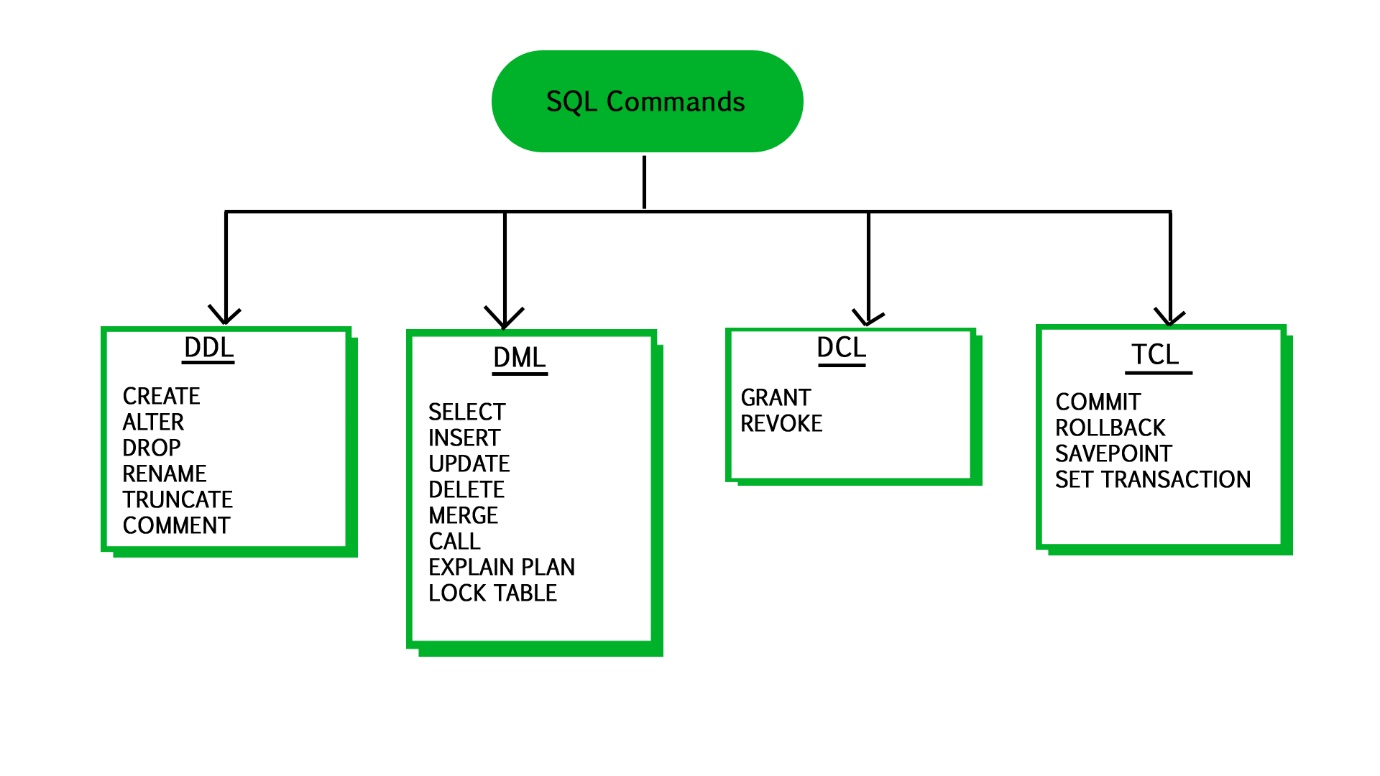
When it comes to Database Management Systems, there are 4 key languages that allow valid expression of queries and structure data stored within a database.

They are:

1. **Data Definition Languag**e: Defines the structure of data and ensures that data is stored in ways patterns can be identified. DDL statements enable definition of the metadata of a database. Creating schemas, tables, indexes and constraints within a database are all done using DDL.
2. **Data Manipulation Language**: Allows accessing and manipulating data that is within a database. Whenever a machine requires accessing something stored in a database, it must do so through an implemented layer of DML.
3. **Data Control Language**: This language has a lot of similarities with DML, but the key difference is that DCL allows the control of permissions to a database, ensuring access hierarchies and also provide the functions of DML. Execution of DCL statements is heavily transactional, and also consists of rollback parameters.
4. **Transaction Control Language**: Persisting changes made to the database is done solely by this language. Each particular statement that is executed to affect the database can be visualized as a “commit” and saving or undoing these commits is what TCL does.

The Structured Query Language incorporates the implementation of all these 4 languages used in Database Management Systems. As such it is highly important in the process of using databases for. The following image shows how each of these 4 languages have been envisioned in SQL using commands.

Figure 1.0, [www.geeksforgeeks.org](http://www.geeksforgeeks.org), shubham\_tyagi4, 2016/05/24



## Choice of DBMS for discussion purposes: Microsoft SQL Server

Being one of the most popular Relational Database Management Systems, MSSQL was first released in 1989 under the version number 1.0, by Microsoft, in conjunction with Sybase. MSSQL was extremely similar to the releases of Sybase during the initial stages, however though this partnership between Microsoft and Sybase dissolve somewhere close to a couple years later. Microsoft however, made sure to retain its rights to MSSQL.

Ever since then, the heights MSSQL has reached have been tremendous, with different levels of MSSQL that a particular company can subscribe to, it’s become the industry recommended and most popular solution. As such, given I had previous experience with it and also given the exposure it has had notably since the 1990s, it makes the perfect candidate for evaluation if need be.

## Data Corruption, a problem every developer faces

SQL Server is a platform used for business stability. However, just like any other database, it's prone to corruption. It is the damaging or destruction of data due to some sort of external factor.

There are 3 most common errors that occur as a result of data corruption:

1. SQL Error 5172
2. SQL Fatal 823 Error
3. 8946 SQL Server

The only solution in such a situation will be to restore a backup that has been previously stored, as such making backups is important.

### SQL Error 5172

The SQL Server saves its physical information during a primary file that has information as per the pages. The first page holds the knowledge of the MDF file header, that is named a header page. It consists of assorted info concerning the information like the dimensions of the file, signature, etc.

During session of attaching the MDF in SQL Server, a frequent error encountered by users is error 5172. This generally occurs when the MDF file becomes unhealthy or damaged. Once this error occurs, the information of the header file has already been mismatched, making it difficult to access the data.

This error usually occurs due to:

1. Bad shutdown of the server in which SQL Server is in.
2. Malicious virus attacks
3. Sudden shutdown of SQL server.
4. Broken server hardware

Recovering from this error can only be done through the expertise of a Database Administrator or a System Admin that has expertise in the data sector.

### SQL Fatal 823 Error

SQL users use Windows system calls for the execution of I/O operations. Upon the completion of I/O operations, SQL verifies for any error related to the API calls. If these API calls is incompatible with the OS, error 823 happens in SQL Server.

This error message consists of the following information:

* Whether the I/O operation is a write or read request
* The error code of the Operating System and error description

The error 823 message signifies that there's a problem with the underlying storage machine hardware or a driver that's within the path of I/O request.

Users may face this error when:

* There are contradictions in the file system.
* If the database file is corrupted.

The only way to recover from this type of error is to move to a compatible MS Windows installed machine, fitting the System Requirements for the solution to run.

## Stackoverflow – where developers learn, share and build careers

Stack overflow, inspired from the commonly known computer programming error, is a question and answer website that is very famous among any developer. It is the goto source for all the needs and questions of a developer. Whether you’re just starting out or have been in the industry for some time, you will definitely come across stack overflow.

In fact, whatever issue you may have while programming, if you google for solutions the first source that will popup will be from stackoverflow. Stackoverflow is a branch of the Stack Exchange network and was created by Jeff Atwood and Joel Spolsky.

**I will be using the Stackoverflow developer survey 2019 results as my secondary research quite extensively.**

## Object Oriented Programming – Imperative programming, an example of the opposite of declarative programming

Object-oriented programming (OOP) is a model that revolves around objects rather than a particular set of data or a predefined logic. Programming has always been viewed in the past as a way to simply give realistic substance to logic in such a way for a particular set of or single input(s), an expected output will be reached.

The need for OOP arose due to a set of developers finding the restriction of programming just being tied to logic, and instead required the ability on “how” to define the data that will then be manipulated by logic.

Ever since this particular model was implemented, the way programming was done, changed.

The first step when it comes to OOP is what’s factually known as “data modeling”. In this particular step, the programmer should identify what his data objects are and what types of relationships each object has with the rest of the objects.

After data modeling has been completed, the sum of objects are “generalized” as a class of objects, which will attempt to basically give a clear understanding or a labeling of some sort that will identify what type of data each object contains, which gives a direct idea on what type of logical or otherwise steps can be taken in order to manipulate the class.

This concept called classes will allow a programmer to create custom data types when needed to satisfy the purpose of the programmer that is not already incorporated within the language itself.

Classes also ensures reusability not only from the file it is defined but also in countless other OOP projects which makes distribution of OOP codebases very easy.

Custom or technically known as an abstract data type allows a programmer to employ data structures effectively to organize and manipulate data.

These “logical steps” are referred to as “methods” in OOP. Methods are employed by classes in order to grant the programmer to effectively and logically pursue any necessary requirement. You could say that methods grant the programmer an interface to interact with the objects in the class.

Classes play a huge role in OOP; one big factor that puts OOP apart from the rest is “inheritance”. OOP allows subclasses to be created from a class. Subclasses can be thought of as children to a parent class that can “inherit” the parent’s attributes.

Inheritance allows a programmer to, through creating subclasses from a parent class analyzes the objects of a class, cuts the unnecessary wasting of time on re-inventing the wheel and efficiency during the process of coding.

The OOP model also provides security to its objects through what’s known as “encapsulation”. Encapsulation is concerned with giving access to objects only when there is a necessity in giving access which in other words means that the likelihood of the objects in a class being corrupted can be reduced effectively.

While corruption is reduced, this is a major boost to the security of the application written using OOP concepts, as it reduces the vulnerability of unauthorized access to data of a class.

# Primary & Secondary Research Data Collection Methodologies

## Sampling existing system documentation

The best way to analyze the existing system is to collect facts from existing documentation rather than from human sources. There are various kinds of documents to collect facts from existing documents. These include:

* E-mails
* Customer complaints
* Suggestion box notes
* Reports that document problems
* Problem performance reviews
* Samples of completed manual forms
* Reports and samples of completed computerized forms

There are 2 commonly used sampling techniques namely **randomization** and the other being **stratification**. Randomization is the process by which you select sample data randomly. Stratification is the systematic process to deduce the variance in the sample data that is taken.

## Research through visiting sites

Research through visiting sites is the process of examining similar problems or the same problem which had been previously solved by other sources through human trial or pure documentation on the part of a company. To solve the problem, the analyst will attempt to visit other company websites to see if they’ve gone through similar problems and see how they came to a solution. By doing so the analyst can attempt to search for databases, books and case studies.

## Observation of the work environment where the problem exists

Observation is the simplest and the easiest fact finding technique, that requires nothing but your presence in the work environment. Through this technique, the analyst participates in the organization, studies the flow of documents, applies the existing system, and interacts with the users.

The key to success through this fact finding technique is to see everything through the eyes of the user than an expert of subject matter. This way problems experienced by the clients of a system can be directly observed. Also, through observation, it is possible to grade the problem whether if it’s trivial or something that must be dealt with as soon as possible by observing the impact the problem causes on the employees of a company.

## Questionnaires

Questionnaires are also one of the most useful fact-finding technique to collect information from not only users but also employees. But the greatest strength of questionnaires is also its greatest weakness. A questionnaire is biased on the user’s perspective, each user may have different levels of expectations and have different levels of knowledge backing their understanding so it might be difficult to come to a conclusion.

This issue manifests the greatest when questionnaires are used in companies with a large variance of sectors with people that have different knowledge bases. As such, questionnaires are recommended for audiences that cater towards one specific knowledge base, this way the bias and the level of expectation with bend towards just one particular path.

Questionnaires are highly effective in the initial stages of providing a solution from scratch than from creating a solution to a problem, this is because you can draw the outline of the expectations, requirements and the scope a user sees as a product.

There are two types of questionnaires:

1. Free-format: Users are allowed to answer questions freely; response is not mandatory. This is ideal for situations when you require reviews, feedback, opinions, possible improvements and experiences.
2. Fixed-format: A predefined format of questions is put, then each user has to answer the question. Response is mandatory as the questionnaire is structured. Multiple-choice questions, rating questions and ranking questions are usually used in this format. Ideal for situations where the analyst is trying to put together the expectations, requirements, scope and the outlined structure of a solution or system.

## Interviews

This is the most commonly used technique, because it is the easiest to catch the problem from the root of the cause. Interviewing are usually in the form of face-to-face, where you ask someone questions that will provide useful insight. This insight is gauged in order to understand, verify, clarify and perceive facts about something.

It is in the best interest for a system analyst to conduct interviews than have a second or third party do the interview for them, because during the interviews clues left upon by the gestures of the interviewee are very important. This is because, psychologically humans communicate 70% through gestures and body language. As such, if the analyst is present real time to see this part of the communication, the facts that are henceforth produced will be much more accurate.

However though, if the analyst is a poor communicator or if the interviewee is a poor communicator, the results from the interview will not be as perfect. Communication is key in the interviewing research technique.

## Prototyping

Prototyping is a specialized research technique that is used to collect the requirement part of the system or solution. Prototyping is process by which a small part of the working model of the existing system is sampled, and then observed, interacted with and then understood. This is also used in the design stages of the software development cycle.

Prototyping technique’s best can be extracted after a confident knowledge base has been found and fathomed. A several list of facts must have been already be found before using prototyping. This way you can use prototyping to gauge the accuracy of the facts that have been found previously.

# 

# Merits, pitfalls and limitations in above research data collection methods through compare & contrast

In this section similar research techniques will be compared and contrasts will be drawn. I believe, there’s no point in comparing or contrasting unlike research techniques because the end object of certain research techniques is different in simpler terms, their purposes are different. So comparing prototyping with questionnaires seem pointless, a better comparison is prototyping with sampling.

The first sentence of each comparison/contrast will be a justification as to why I chose to compare or contrast them. To justify that they have similar end goals or purposes and as such are eligible to comparison and contrast.

## Prototyping vs Sampling existing documentation

Both research techniques use the same core, they sample different things to understand.

1. Sampling is a research technique that can be used in the earlier stages of analysis, prototyping must be used in the later stages.
2. Sampling is usually used to gather facts, prototyping is used to gauge the accuracy of facts that have already been found.
3. The facts found through sampling may be outdated, but prototyping provides accurate upto date facts based on a presently existing system.
4. Prototyping requires a certain level of knowledge and experience to perfrom, but sampling does not, it just needs the ability to read.
5. Prototyping needs a working modal to exist which is rare than finding system documentation for sampling. So the likelihood that you will use sampling is higher than prototyping.

## Research through site visiting vs observation of work environment

Both research techniques require little to no investment of resources. Their core is observation, and the end goal is to understand.

1. Site visiting does not require cooperation of an audience while observation does.
2. Site visiting will take much less time to perform than through observation.
3. The facts gained through site visiting is concrete. They’ve been tested and observed countless times. Observations in the work environment may happen only just once and never again.
4. Observation of the work environment helps understanding the impact a particular problem has on the working environment; this is not possible through site visiting.
5. Site visiting is not always successful; you are not guaranteed to find solutions or facts that are related to your problem. Observation however will definitely help you understand the situation and come up with a solution on point.

## Questionnaires vs Interviews

Both techniques rely on an outside audience, the facts that you obtain through these techniques may or may not be biased.

1. Questionnaires do not require the analyst’s presence, while interviewing requires not only your presence but an interviewee as well.
2. Good communication is a mandatory requirement of interviewing; questionnaires does not have this requirement.
3. Interviewing may take long periods of time on large audiences, questionnaires can be collected within a couple hours if you wished to.
4. Interviews are ideal in situations where feelings, experiences and what not are a strong requirement to understand the problems. Questionnaires fail in times like this, because feelings can’t be expressed accurately through words.
5. Questionnaires for large audiences require a budget enough to design the questionnaire, test it and then print it. Interviewing has a low budget requirement.

# Justification for the choice of data collection methods

**For the primary research**, I have chosen to use a 5 part questionnaire and the reasons as to why I have chosen **a questionnaire** are:

* Allows to directly gauge personal opinions and experiences in an indirect manner, which allows the participant to psychologically feel safer and less nervous opposed to something like an interview or observation.
* Ensures that information that is gathered are more scoped to the objectives of the research.
* The least cost, ethically proper and time saving method to get personal opinions are a questionnaire.
* There is no interference by the researcher on the participants as they can take as much time to think and give a personally satisfying answer to the questions, hence improving the quality of the information gathered.

**For the secondary research**, I have chosen to analyze the **research results** of the **Stackoverflow Developer Survey of 2019**, and the reasons are:

* Stackoverflow is a medium for developers, and my audience are developers, the research was not only conducted to identify trends in the industry but also sensitive things like gender, age ranges and what not which I do not store. As such I am not responsible for the security of the data.
* The research results of the developer surveys are publicly online for use and analyzing as long as the proper credit is given. Several other industries and developers have utilized these research results for their personal projects or researches.
* Both quantitative and qualitative analysis can be conducted as the results have been released as bar charts with percentage values.
* Over 190,000 developers have participated in both of these developer surveys in total and as such the conclusions I take from them must be quite valid.

# Primary research – Questionnaire

## Preview of the questionnaire

The questionnaire I distributed to the internet both over stackoverflow and a more of a softer social site called reddit, had 5 main questions that were constructively created in order to gauge the results such as to achieve the objectives I’ve laid out as the goal of this research.

These were the questions that were asked and the reason why I asked them:

1. **Do you consider yourself a beginner or an expert in the programming industry? (3+ years of experience would make you an expert by default)** – In order to classify my responses to 2 categories so that the opinions of actually experienced developers don’t fluctuate with the responses of the beginner developers. If there were some fluctuations the information will be not be normally distributed to the 2 sections.
2. **Do you think that SQL is difficult to learn? Why/Why not?** – To understand the participant’s opinion on my research problem, whether if it is actually difficult or not difficult to learn SQL as a whole and what they personally think is so difficult about it.
3. **Have you had sufficient experience with SQL? (Knowing basics counts as sufficient)** – So as to filter out the opinions of those that have not had sufficient experience with SQL.
4. **What SQL statement(s) have you had the most difficulty with?** – To directly identify what component(s) it is in SQL that makes it difficult to learn.
5. **How would you rate your experience with SQL in production applications from 1 to 10?** – Gathering quantitative information makes it very easy for me to come a step closer to answering my research problem and grasp my objectives.

## Outcomes & conclusions

After the questionnaire stayed on the internet for over a week, 54 people had participated in the questionnaire and out of that 54 around 39 people had the type of responses that I was looking for. The other 15 responses were either spam or those that I decided to filter out depending on the answer of the 3rd question in the questionnaire. So the outcomes will only consider the responses of **39 developers.**

**Outcome #1 – 76% (29) of the respondents were beginners and 24% (11) were experts.**

This means that I have nearly a 1 to 3 ratio of experts to beginners answering my questionnaire. As such I have sufficient participants in order to draw mid-range valid conclusions from their answers.

**Outcome #2 – 81% (23/29) of the beginners said SQL is difficult and 15% (2/11) of the experts said that SQL is difficult.**

This means that my research problem is valid to a point, and that there is in fact beginners struggling with SQL, however though as it seems it’s become obvious that the experts don’t seem to have as much trouble. But 11 is quite a small number, if this number were higher I am sure that there’d be a much higher percentage among experts.

**Outcome #3 – 72% (39/54) had basic knowledge of SQL**

This question was more of a way to filter out responses and increase the quality of the information than a way of gathering information. It allowed me to filter out the responses I didn’t want, much easily.

**Outcome #4 – Generally among the respondents, it seemed apparent that JOINs and nested queries were the factors that made SQL more difficult. However though, most of those respondents also agreed that they’re very powerful features and as such the difficulty is a rich compensation.**

Essentially means that there is in fact a few components in SQL that make SQL difficult. 2 of the identified statements have been shown to be JOINs and Nested Queries.

**Outcome #5 – 67% of the ratings were above 7**

This means despite the difficulty concerns of SQL there is, that the difficulty is a calculated risk. The power and flexibility of declarative programming certainly cuts down the difficulty, which is probably why most of the ratings were above 7.

# Secondary Research – **StackOverflow** Development Survey Results 2019 outcomes & conclusions

All credits goto the respectful and wonderful community of Stackoverflow, and every piece of information that I’ve utilized in this research are credited towards Stackoverflow. I claim no ownership or rights over the information, I will only comment on the survey results and grasp conclusions.

## Outcome #1: SQL – The #3 most popular language in the world

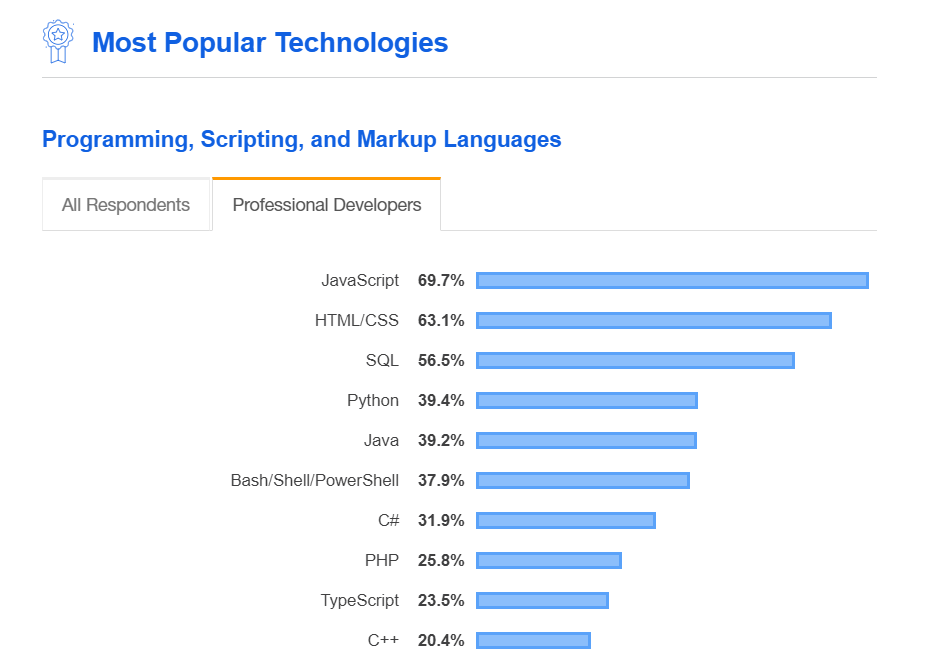


Figure 1.1, insights.stackoverflow.com, 16/06/2019

As you see, 56.5% of all the respondents of the survey has voted SQL to be the 3rd most popular language in the world. This answers our 2nd research objective “If SQL is demanded in the job industry”, the outcome #2 will back this claim. SQL has triumphed over imperative languages like Java for example, who most developers don’t have as much difficulty to master the basics. This essentially means that the place SQL has in this world cannot and will not be replaced by any other technology.

## Ouctome #2 – SQL is #18 highest paid language at $57k annually

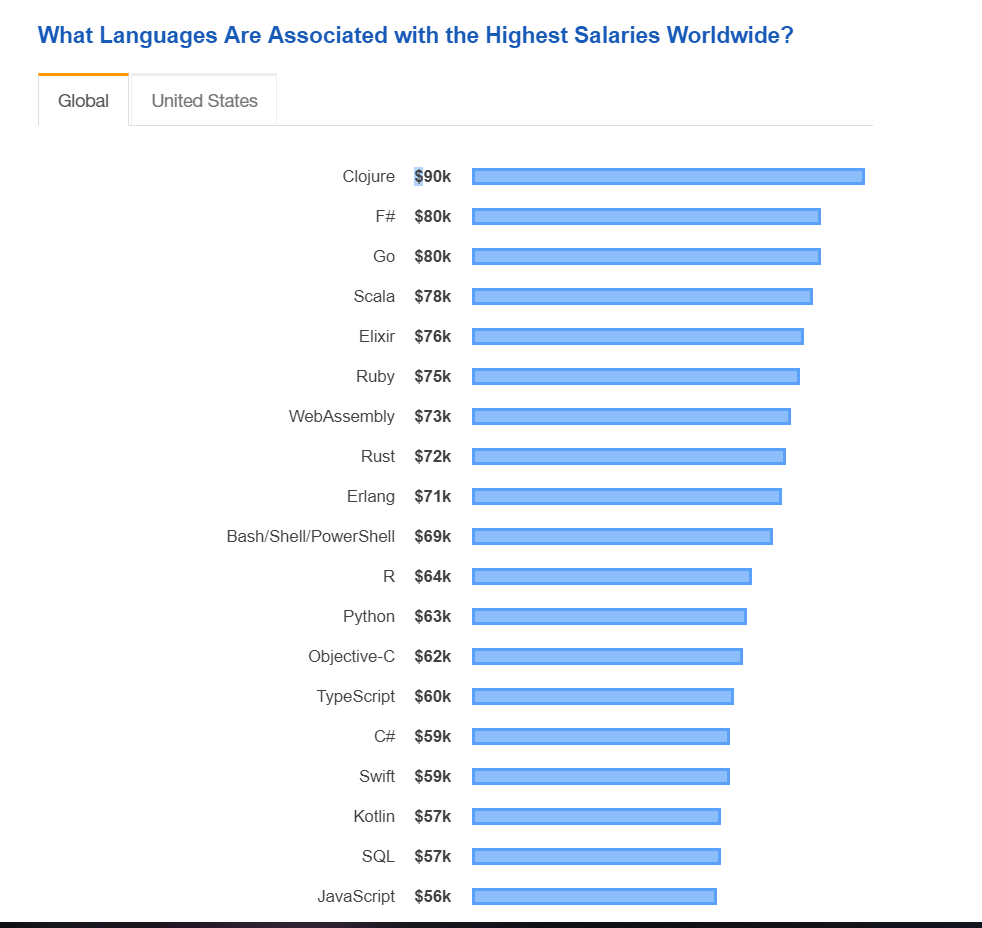


Figure 1.2, insights.stackoverflow.com, 16/06/2019

Further backs our research objective of if SQL has a job demand in the world, SQL triumphs over the #1 most popular language in terms of salary paid annually. This further backs the value of SQL in this world. $57k per year is roughly $4.75k per month which is a very high salary. The reason for this must be because SQL is largely used in database programming and data science. As there is an expected increase in the demand of data science experts in the near future, it’s safe to say that this salary will increase.

## Outcome #3 – 64.1% votes SQL as #13 most loved language while 35.9% votes SQL as #15 most dreaded language

This outcome is quite the contradiction, isn’t it? Essentially more people love the language than finding it dreadful, but however it is definitely dreaded by quite a few people. Why could this be the case? Simple, it’s difficult to learn it or master it. This outcome mustn’t be looked at as saying that SQL is not difficult to learn just because more people love it, reason being that there’s 35.9% people voting SQL as #15 dreaded language.

If so what is the conclusion that can be gained from this? Two conclusions can be taken from this outcome:

* There’s people that love the language this is for certain, as such this proves that the language is very powerful and satisfies the needs of the developers, **this essentially proves that the language has significant influence in the job industry.**
* There’s also people that dread the language, this means that the language is actually difficult despite it being lovable. **As such this proves that there must be pitfalls in the process of learning or working with the language.**

Figure 1.3, insights.stackoverflow.com, 16/06/2019

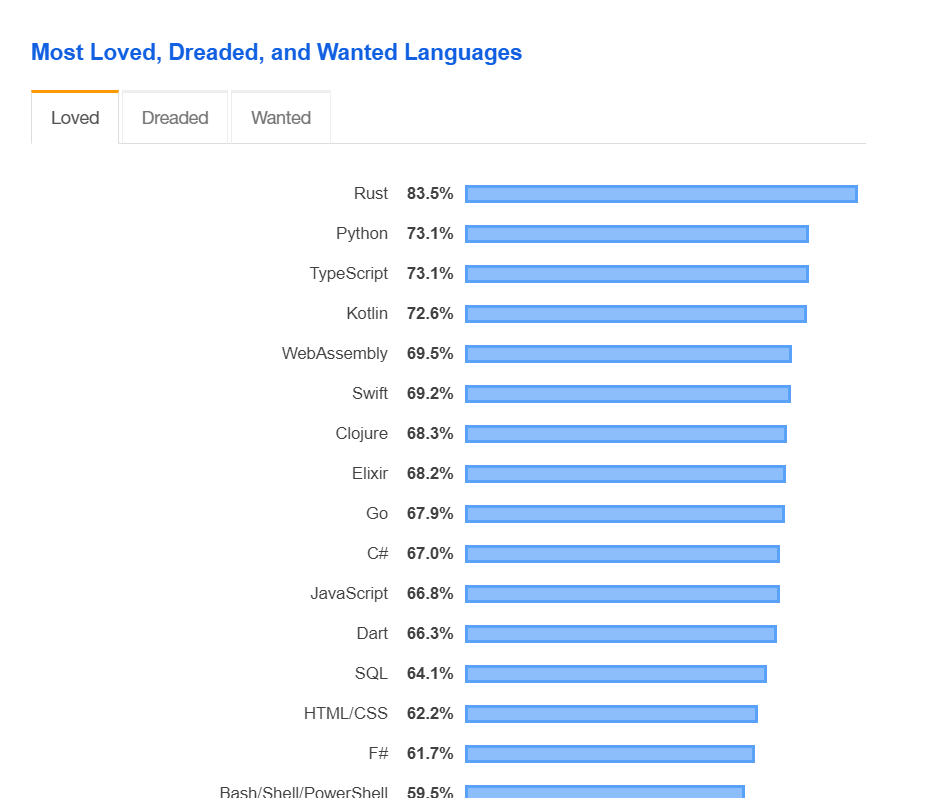
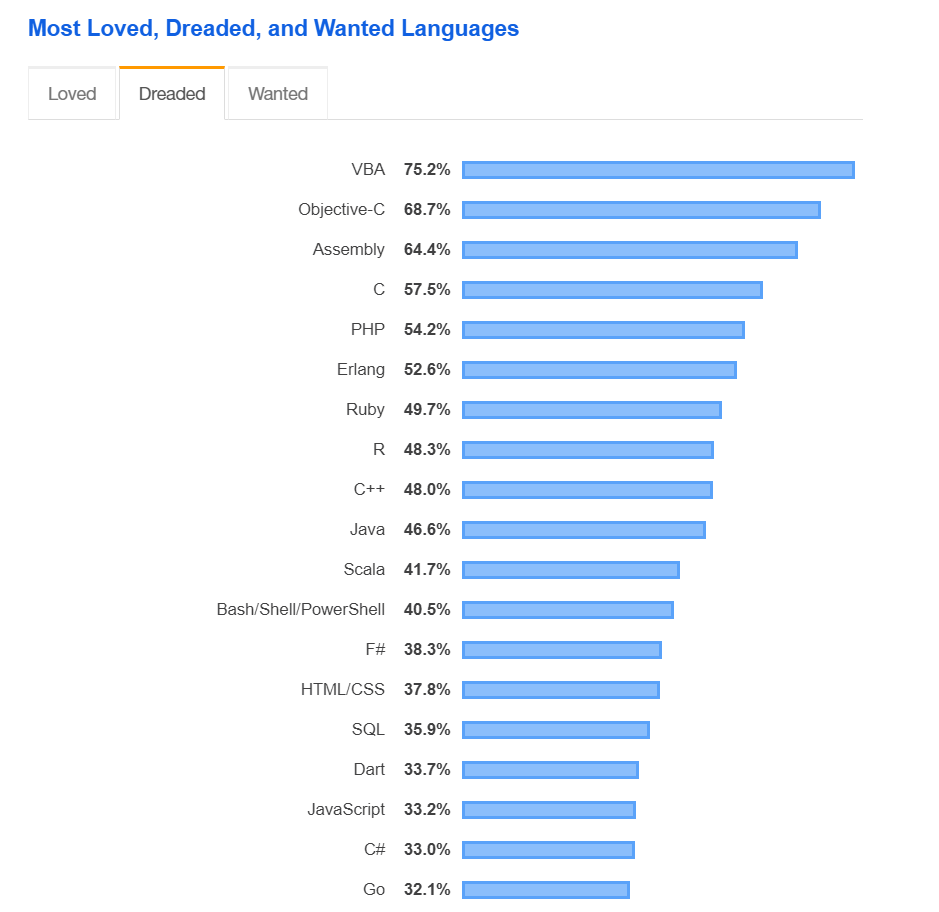


Figure 1.3, insights.stackoverflow.com, 16/06/2019



# Final Conclusions

The research ended on the 5th of June 2019, 4 days after the actual planned date of finishing the research. Over the course of the month, the primary and secondary research methods were all utilized in such a way to find the answer to the research question and hopefully satisfy the research objectives. As such the following are the main conclusions that could be made from all the outcomes of this entire research:

* SQL is a very powerful and popular declarative language in the world, mainly because it lacks conditional logic. It is more of a language that says what to do than how to do it, as such for contexts and purposes like data science, machine learning etc. it is irreplaceable.
* SQL cannot and will not be replaced by any other declarative language for database programming, the secondary research proved this with very large numbers that there is no competitor much less that can provide the functionalities of SQL.
* SQL is in fact a difficult language to learn, notably from the primary research outcomes saying that Nested queries and JOINs are culprits.
* However, the difficulty of the language is compensable due to it being extremely powerful. The 2 culprits nested queries and JOINs are very powerful features in SQL. As such it is reasonable to say that they are quite difficult to utilize, satisfying the 1st research objective.
* SQL is a very demanded language in the job industry as proven by the secondary research outcomes where the monthly salary came to $4.75k and SQL being the #3 most popular language in the world, both these statements contribute to SQL being a demanded language. As such it is reasonable to say that the 2nd research objective is satisfied.
* Both the fact that SQL is demanded and is a difficult language means that it is safe to assume that there must be quite a few pitfalls in the learning or mastering process of SQL. As such it is reasonable to say that the 3rd research objective is satisfied.

The conclusion to the research question “**Is SQL difficult to learn**” is “**Yes, it is difficult to learn SQL but the power, features, job demand and popularity of the language overwhelms the difficulty**”.

# Reflection on research methodologies and possible future improvements

Simply, I’d say that the research was very successful and I learnt quite a lot about task/project management, the different levels where people stand at and subtle things in the development industry that most developers aren’t aware of. This was my first attempt at doing a research on something that is computer based and there were so many struggles amidst all of that however, each step I took gradually grew closer and closer to satisfying the research objectives. In the end I answered the research question and solved the research problem. Any member of the intended audience who wishes to read my research documentation will be capable of learning quite a lot.

Using questionnaires and the stackoverflow development survey results were quite difficult but well worth it. Specially the usage of the survey results, it had already condensed the information from lots of respondents which allowed me to effectively conduct quantitative and qualitative analysis on the research data. The questionnaires proved useful for the primary research method, however improvements can be made. I’d say that if I were to research this same topic again, I’d combine questionnaires with live interviews so I can get a more information rich set of outcomes and conclusions.

## Alternative research methodologies for improving

Some alternative research methodologies I could have taken instead would be interviews and use sampling methods of some type. Interviewing the students who had difficulties in the institute where I got the inspiration to start this project would have added to the total number of participants for this research. The sampling methods could’ve somehow allowed me to find more indirect connotations between the research objectives as opposed to those that I have taken from the secondary and primary research. Basically, these 2 research methods opposed to or with the 2 research methodologies that I have already utilized might have bought more context and richer information to use as the basis of this research.

## Lessons learnt

This research lasted a month and I learnt a lot of things and improved on a lot of things that I already possess, some of those lessons are:

* SQL has a very high job demand around the world, I did not ever think that a declarative language that was used alongside some imperative language would have a such a high direct demand.
* More beginners seem to struggle with SQL than I originally thought, in fact there are a lot more misconceptions like SQL is not needed in fullstack web applications or that NoSQL is superior to SQL because it provides a more OOP or a functional approach to database programming.
* Pacing yourself during a research is very important, I tried to tackle multiple things at the same time which resulted in a delay in my original plan.
* Thinking about costs, ethics and security is very important in a research that includes living participants.
* Structuring your research skeleton is crucial as this allows you to understand what has to be researched and how it has to be researched.