Master SQL for Data Science  
[Beginners Guide]



**Anish Mahapatra**

Contents

[Introduction 4](#_heading=h.gjdgxs)

[What is SQL (Structured Query Language)? 4](#_heading=h.30j0zll)

[Why SQL for Data Science? 5](#_heading=h.1fob9te)

[Need for SQL in Data Science 6](#_heading=h.3znysh7)

[Key elements of SQL for Data Science 6](#_heading=h.2et92p0)

[Using SQL for Data Science 7](#_heading=h.tyjcwt)

[Data Understanding 7](#_heading=h.3dy6vkm)

[Business Understanding 7](#_heading=h.1t3h5sf)

[Profiling Data 8](#_heading=h.4d34og8)

[Start with Select 8](#_heading=h.2s8eyo1)

[Test & Troubleshoot 8](#_heading=h.17dp8vu)

[Format & Comment 9](#_heading=h.3rdcrjn)

[Review 10](#_heading=h.26in1rg)

[SQL for Various Data Science Languages 10](#_heading=h.lnxbz9)

[SQL for R 11](#_heading=h.35nkun2)

[SQL for Python 11](#_heading=h.1ksv4uv)

[SQL with Hadoop 12](#_heading=h.44sinio)

[SQL for Spark 12](#_heading=h.2jxsxqh)

[Prerequisite 13](#_heading=h.z337ya)

[Getting Started with SQL 13](#_heading=h.3j2qqm3)

[Creating Tables 13](#_heading=h.1y810tw)

[Adding/Inserting Data to The Table 14](#_heading=h.4i7ojhp)

[Retrieving Data 14](#_heading=h.2xcytpi)

[Adding Comments to SQL 15](#_heading=h.1ci93xb)

[Filtering, Sorting and Calculating Data with SQL 16](#_heading=h.3whwml4)

[Filtering data and Sorting Data 16](#_heading=h.2bn6wsx)

[Aggregating/Calculating Data 17](#_heading=h.qsh70q)

[6 SQL Tricks Every Data Scientist Should Know 17](#_heading=h.3as4poj)

[Order of execution 17](#_heading=h.1pxezwc)

[Master the “GROUP BY” clause in SQL 18](#_heading=h.49x2ik5)

[Joins 18](#_heading=h.2p2csry)

[Nested queries or Subqueries 19](#_heading=h.147n2zr)

[Stored Procedures 19](#_heading=h.3o7alnk)

[Working with Indexes and Null Values 19](#_heading=h.23ckvvd)

[Steps to Master SQL for Data Science [From Scratch] 19](#_heading=h.ihv636)

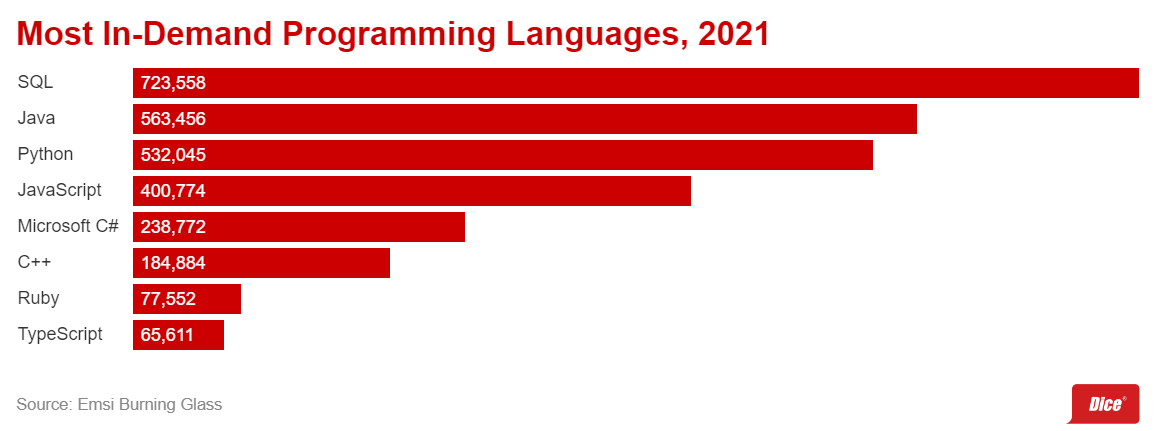
[Conclusion 20](#_heading=h.32hioqz)

[Author 20](#_heading=h.1hmsyys)

# Introduction

There have always been two major approaches that most of us take up while learning a new subject. One is the classic method of systematically learning all the concepts by going step-by-step from basic to advance levels. The other one is *soaking everything one night before* the interview or an exam. For Sequel data analytics, we may have seen many instances where our colleagues start to rote learn to get a hang of the topics and concepts. This method might get you through your interview, but in the long run and in your daily work you will be back to struggling and going around googling the same stuff over and over again to perform simple tasks. In case you are interested, you can refer to the [best Bootcamp data science](https://www.knowledgehut.com/data-science/data-science-bootcamp-training).

The road you will have to take to master SQL programming specifically for data science, which still many employers seek as the most in-demand skill, is the question we will be addressing in this article. If we were to look at the trends of the skill across the past few years, we will be able to find SQL always being a part of the top 10 programming languages. As seen here in the last year’s most in-demand skill was **SQL**. This article will act as a SQL guide for beginners which will help you understand how does SQL work and how you can master it for Data Science.



We can also look at [data science course details](https://www.knowledgehut.com/data-science-courses) here. So let's start through the approach of understanding what SQL for Data Science is through its essence and the step-by-step process of how you can get started with learning and mastering the skill, apply it for Data Science add also it to your armoury of skills.

# What is SQL (Structured Query Language)?

SQL or Structured Query Language or ess-cue-ell or even SEQUEL, or whatever we call it, let us first try to understand what SQL is and get a general feel of all the applications. Sequel programming languages is a popular term as well.

*Structured Query Language (SQL) is a standardized programming language that is used to manage relational databases and perform various operations on the data that is present in them.*

With SQL you will be able to create, insert, alter, update and filter data to any database format that you will see fit or optimum for further usage.

Here, I would be introducing new terminologies to determine the applications of SQL based on the set of commands or statements we use.

* **Data Definition Language (DDL):** The commands that are a part of DDL are the ones that will be used for defining a database schema. DDL consists of commands that are used to create, modify and delete database structures. These commands are generally used by the Database admins and lesser by the general users. Ex: CREATE, ALTER, DROP, TRUNCATE
* **Data Query Language (DQL):** These statements used within DQL are used for querying the data from the schema. DQL includes the **SELECT** command. This command allows getting the data from the database to perform operations with it.
* **Data Manipulation Language (DML):** This section of SQL consists of the most number of SQL commands as you will be using them more to manipulate the tables and the data within them. Ex: INSERT, UPDATE, DELETE, LOCK, etc.
* **Platforms for SQL**: We can also find that platforms like Hadoop and Spark which are popular services for handling and working with Big Data, provide an extension for querying and manipulating using DDL, DQL and DML commands of SQL
* **Data Control Language** **(DCL):** Commands included in this section are GRANT and REVOKE which are essentially used for providing rights, permissions and other controls within the database systems. We will also go over what is a relational database. Mainly the database admins would be dealing with DCLs

The front-end to the back-end connection is established through the Client/Server language, which is SQL and is extensively used in the data science industry. A good way to learn the fundamentals is with [data science with python courses](https://www.knowledgehut.com/data-science/data-science-with-python-certification-training).   
  
Now that we have gotten an idea of the definition of SQL and the applications around it, let's get into the next question, which forms the heart of this article, Why is SQL used for data science?

# Why SQL for Data Science?

Reiterating the above point of SQL data science being the most in-demand skill and relevant even after four decades of its existence, tells us a lot about how it offers several key features and advantages over the alternatives. It can also be noted how almost all of the data science job postings that we see today on different platforms list SQL as one of the primary skills. In another survey from the 2020 LinkedIn top 10 Startups from India list, 7 have SQL listed as one of their **Most Common Skill.**

Let's look at a few compelling examples or applications of SQL within Data science to understand the need and importance of SQL in data science

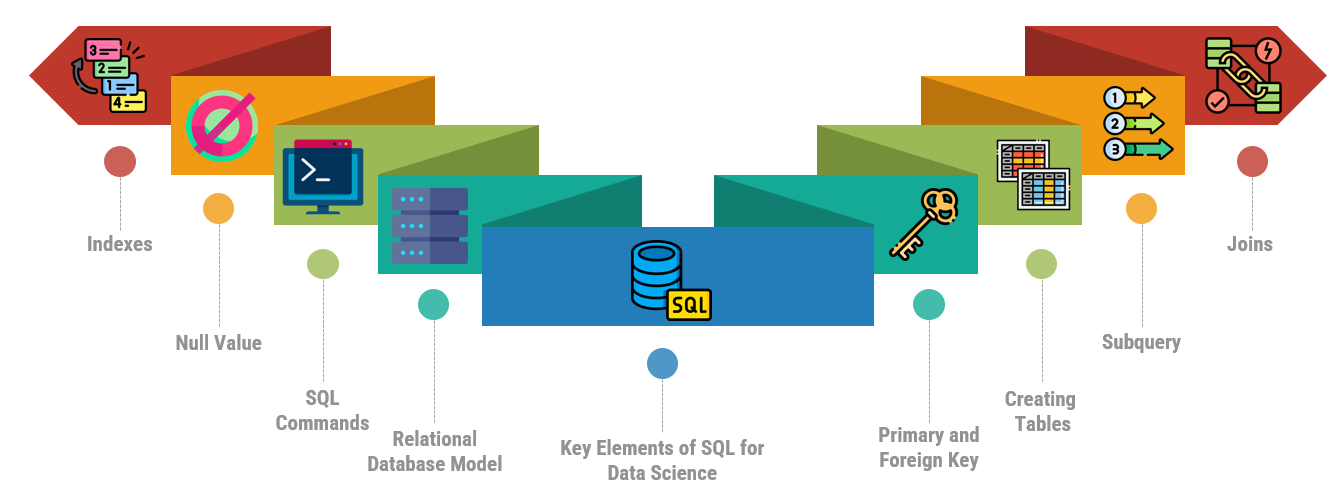
## Need for SQL in Data Science

The following points highlight the need for SQL in Data Science.

* We can define Data Science as simply an all-around study of the data itself. To enable this, the first step is to extract this data from the databases, where SQL comes into the picture. Most of the companies we see today find Relational Database management as a crucial part of their data management ecosystem. SQL allows us to perform all the actions on these databases such as defining, manipulating, creating, querying the databases and many more.
* We can also find that platforms like Hadoop and Spark which are popular services for handling and working with Big Data, provide an extension for querying and manipulating using DDL, DQL and DML commands of SQL.
* For all the data stored in Oracle, PostgreSQL and Microsoft SQL Server which are examples of a few popular relational databases used by large companies, we would need SQL as well
* SQL also becomes a necessary tool for carrying out data preparation and wrangling actions. Therefore, to deal with various Big Data tools, we will be making use of SQL

Dealing with big data and relational databases using SQL may sound a tad bit overwhelming, but don't worry, we’ve got you covered with the Key elements that you will require to get into the groove of understanding the above few examples.

# Key elements of SQL for Data Science



#### Image by [Author](https://www.linkedin.com/in/anishmahapatra/)

These are a few concepts that we believe will be essential for using SQL within Data Science.

The topics mentioned here will help you focus on areas that are crucial to know. We will be looking into these topics as we keep going further into the article.

# Using SQL for Data Science

Now that we have understood the importance and the need for SQL in data science, let's quickly move into the section on **how we go about using SQL for data science.** Let us here approach this in the same format one would usually approach a data science problem. I have here included a flow chart to showcase the steps that we are going deep dive into.



#### Image by [Author](https://www.linkedin.com/in/anishmahapatra/)

## Data Understanding

One of the most crucial aspects of being able to solve a problem in any domain/industry of data science is to understand the data to the fullest. In most companies, there is an “80/20” rule that is followed whereas it is seen that 80% of a data scientist’s valuable time is spent simply finding, organizing, and cleansing the data, only 20 per cent to performing actual analysis on the prepared data.

So this stage here talks about how one would need to spend a significant amount of time to understand the in and outs of the data, conducting multiple sessions with the team and the stakeholders to dig deeper and gather a solid understanding of the data.

## Business Understanding

When we mention that we would need to conduct multiple sessions with the business/stakeholder to understand the data, there is also another aspect that we will need to consider here i.e. the **business understanding**. The understanding of data is incomplete without the business aspect, as we will be missing out on crucial contexts that we can acquire through the business.

Here, let us look into an example of a business problem to find the annual revenue, profits and losses and the year-on-year growth of stores across the country.

The problem, even though seeming straightforward, having interactions with the business/stakeholders will give us directionality as to what products to focus on, what stores to put more weightage on, and many such details which enable a data scientist to shape their analysis. The two stages above help us to fill in the blanks within the queries that we will want to work on as we progress with data and business understanding.

## Profiling Data

One might think about what profiling data mean and how it fits in here. We can define it as the process of analyzing, examining, reviewing, and summarizing data sets to gain insight into the quality of data.

* **Data quality** can be measured using the factors of accuracy, completeness, consistency, timelines and accessibility of the data. To move any further with the actual analysis of the data, it is necessary to conform to a high quality of the data. A good way to understand this is with the primary key in SQL(A SQL server primary key is any column that possesses unique values for each record. A primary key example in SQL can be an employee\_id column in a table of employee details)
* **Data profiling** also involves reviewing the source data to be able to understand the structure of the data, its content and the interrelationships between them. This process of data profiling benefits any organization two-fold. It helps them to gain a high-level view of the quality of the data and also helps them identify potential data projects for the future.

Once the data profiling step of managing the data with SQL with all its quality checks and ensuring the data is at its highest standard, the next step is to query the data using SELECT and FROM.

## Start with Select

As the title suggests, post completing the above three sections, you will be starting with SELECT. The **SELECT** command in SQL is one of the easier places to start your analysis. As the data would be new to look at and understand, one can begin by starting simple with a single table first. Once you start getting comfortable with the analysis, more data and tables can be added to it and you can go back from there.

This leads to the next step of looking into the testing and troubleshooting of the queries that we create.

## Test & Troubleshoot

This step can also be called the QC step or the **Quality Check.** Herein you will be checking the number of records your query will be returning if the query included all the correct filters to get all the required data points to calculate any statistics. This step becomes crucial as getting results is a simpler task, but making sure it is accurate is of more importance.

The next stage is to be able to resolve any queries when we see any discrepancies in the outcomes or the queries themselves. By beginning to dissect the query step by step to identify the source of the issue, we can keep the procedure straightforward.

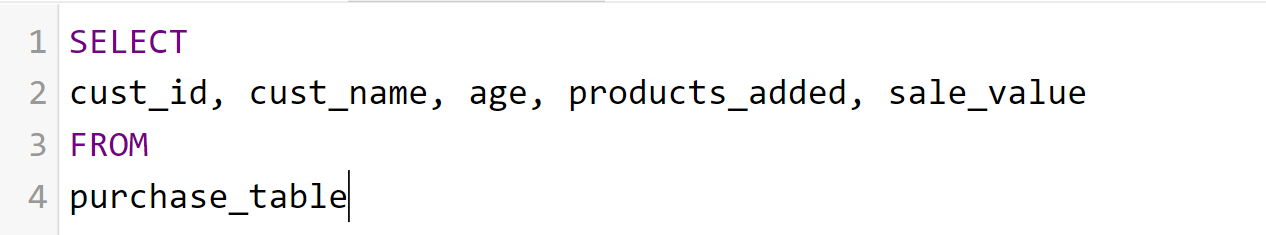
This technique is also useful in that it will enable users to create subsequent queries that are more effective.

## Format & Comment

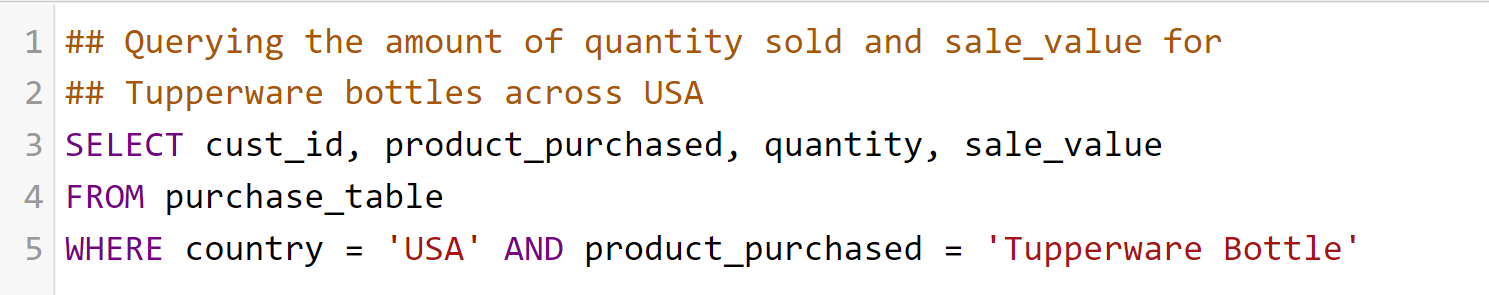
We all know the frustrations of being able to decode a bug from a badly formatted code. It wastes an ample amount of time to debug it and even then we are still left with badly written code.

A bit of digging into this aspect reveals that most programmers want code that simply works and also wants it with a quick turnaround, which makes formatting the last priority for any one of us.

An effective method to avoid that messy code is to maintain two versions of your code, one which you keep to yourself for figuring out the code and the other version which is production ready for anyone else to take it up and also understand it easily. One adequate method to follow in SQL is to capitalize the commands in your statement and the rest of it can follow the lower case. Like the example shown below.



The next key suggestion will also be to ahead and comment on the codes that are written, this will make the code easily readable as the objective is already stated before the code. This can be seen in the example below.



## Review

This forms the last step of our process and one of the most tedious parts of it as well. The entire effort that you put in until now will basically be reviewed from the top to bottom here. This will ensure you have captured all the data understanding and business rules effectively, conformed with the highest quality of data, and formatted and efficiently commented on your code.

# SQL for Various Data Science Languages



#### SQL for various DS languages. ([Unsplash](https://unsplash.com/photos/fPkvU7RDmCo))

Until now we have understood how SQL can be used to effectively understand the data by using queries to define and manipulate the data. But what if we were to tell you that SQL can be used within various platforms as well such as R, Python, Hadoop and Spark.

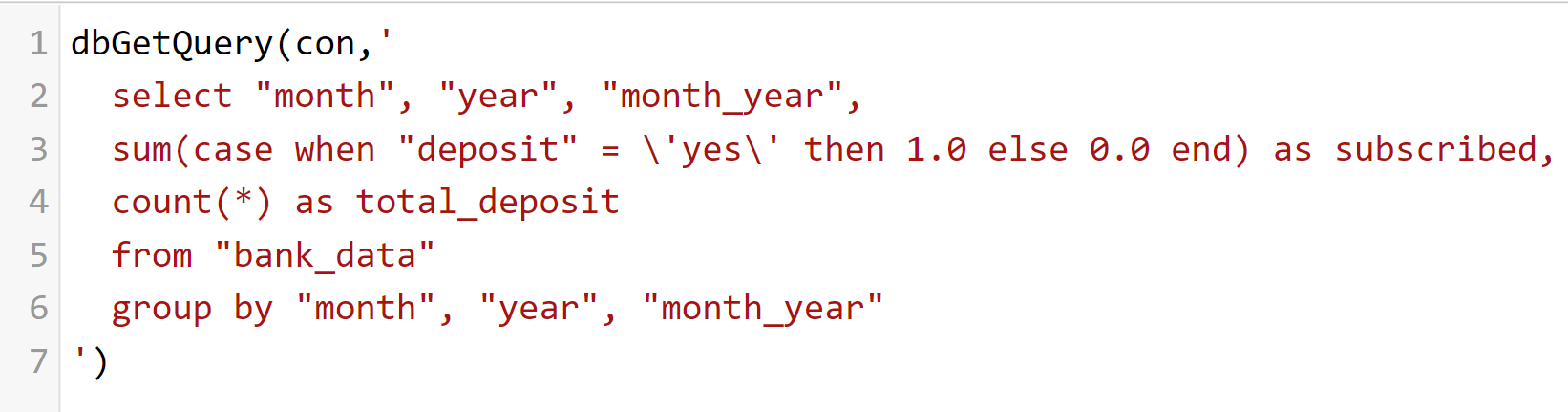
But we also need to be wary of the fact that SQL queries within R and python can only tell us about what has happened in the past and cannot predict any future predictions. So, their uses in these platforms are only limited to fetching, querying and understanding the data.

The general procedure we are going to look at for using SQL within R and Python is to be able to:

1. Setting up SQL,
2. Creating (from scratch) and fetching datasets from databases and
3. To run queries in the same environment of the R and Python workspace

## SQL for R

1. The **DBI** package can be used in the R package for setting up a connection to databases. The DBI package has options for connecting with popular databases. **MariaDB, Postgres, SQLite**, etc. are just a few of them. To see more you can check this [link](https://github.com/r-dbi/backends#readme).
2. Importing **tidyverse** library gives us access to **dplyr, ggplot** and **default datasets,** which we can use to fetch default tables. Or you can even create your own data sets by using the CREATE table syntax and then use INSERT to add in values.
3. The **DBI** package can also be used to query the data by using the dbGetQuery() function. You can do it simply by pasting your SQL code into the R function within a quoted string. This method is also referred to as ***pass-through SQL code.*** An example of this use case can be seen below:



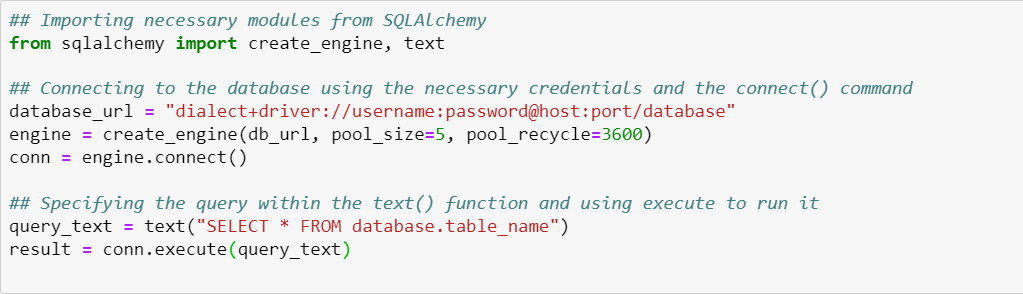
Additionally, you can also use the ***dplyr,*** by writing your code in dplyr with R syntax. **dplyr** has a feature that will convert your R code into SQL queries.

After a little more digging into this prospect, we also found that we can use SQL code in an R notebook. Isn’t that a cool fact you know right now? We’ll let you do the figuring out how to work this last option out.

## SQL for Python

In this subsection, we will be discussing how to manage databases and use SQL for data science with python which can also lead to knowing that it is possible to use sequel (SQL) for data analytics

1. **SQLAlchemy** is another popular python module that lets the user connect to all kinds of databases by providing a unified API. The process involves the creation of an engine where you can specify the characteristics such as the **database name**, **username**, **password**, **host** and **port** for that particular database. More information for setting up connections can be found [here](https://docs.sqlalchemy.org/en/14/core/engines.html)
2. Once the connection has been established to the preferred database, fetching all the tables within that database becomes a possibility or you can also go ahead and create tables on your own. SQLAlchemy also allows us to look at all the tables present in the database and hence helping you to fetch all the tables within it.
3. Two steps need to be performed using SQL Alchemy for you to be able to write SQL queries in python. The first one is to include your query within **text()** and then utilize the **conn.execute()** function to execute your query in that environment
4. A full example of a connection, fetching a table, and execution of a query can be found here,



In the next sub-section, we will be looking at how SQL finds its application in Apache Hadoop and Spark frameworks. But first, a quick context about both of these frameworks. The Apache Software Foundation developed both **Hadoop** and **Spark**, two popular open-source frameworks for big data structures. Each framework has a rich ecosystem of open-source tools for handling, managing, and analyzing large amounts of data.

## SQL with Hadoop

The SQL implementation on the Hadoop platform, known as **SQL on Hadoop**, combines normal SQL-style structured data querying with the Hadoop data architecture. Big data and Hadoop are both relatively new platforms, and few professionals are experts in either. However, SQL on Hadoop makes it simpler to access the Hadoop framework and easier to integrate it into existing enterprise systems.

One of the SQL on Hadoop implementation include the widely popular Spark SQL module, which we shall elaborate in the below section.

## SQL for Spark

A Spark module for processing structured data in Spark SQL. It offers the programming abstraction called the DataFrames and also functions as a distributed SQL query engine. The features also include powerful integration with the rest of the spark ecosystem such as integrating SQL query processing with ML.

For more information on how to use SQL with spark, [this documentation](https://spark.apache.org/docs/latest/sql-programming-guide.html) comes in very handy

# Prerequisite

SQL is a relatively easier topic to start learning, which means it will require minimum prerequisites, to begin with. But, we shall outline a few topics that are good to know before you begin your journey to learn SQL.

In the above segment, we have showcased how to use SQL in R and Python to connect the various popular databases. A good place to start will be to understand the concepts of **Relational Database Management Systems (RDBMS)**. You can explore the following concepts along with understanding what relational model is used for building relational databases, is to get up to speed:

* Table attributes,
* SQL constraints,
* Data Integrity categories and
* Data Normalization

These concepts will give us a clear picture of managing data with SQL and the process of creating tables.

Another important concept that would make it easier for us to retrieve data in a fast manner is understanding how indexing works in SQL.

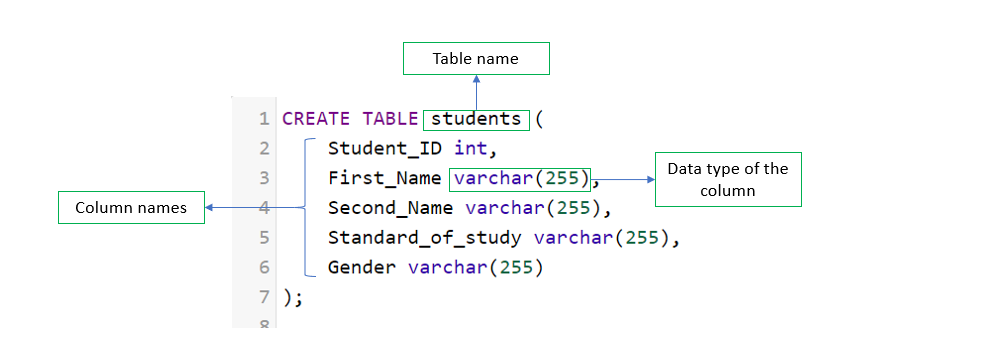
# Getting Started with SQL

Here, in this section, we would like to go ahead and give you a general feel of using SQL. We shall focus on the basics of creating a table, adding inserting into the table and retrieving the same data once it gets created. We shall be showcasing examples through code chunks.

You can use this section to build your first table as well.

## Creating Tables

We shall be using the CREATE command over here and also have a look at its components.

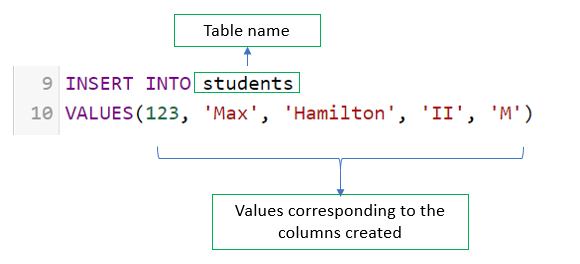


#### Components of a SQL CREATE statement (Image by [Author](https://www.linkedin.com/in/anishmahapatra/))

Executing this above statement will create a new table in your environment. See, it’s that easy. Now go ahead and try out this query over [here](http://sqlfiddle.com/).

## Adding/Inserting Data to The Table

Now the next step is to go ahead and add data to the table that you have just created. This step can be accomplished by using the INSERT command.

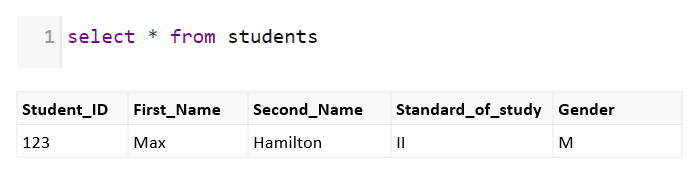


#### Image by [Author](https://www.linkedin.com/in/anishmahapatra/)

One should be careful while using the INSERT statement to insert the respective values into their correct positions w.r.t to their columns assigned during the CREATE statement.

## Retrieving Data

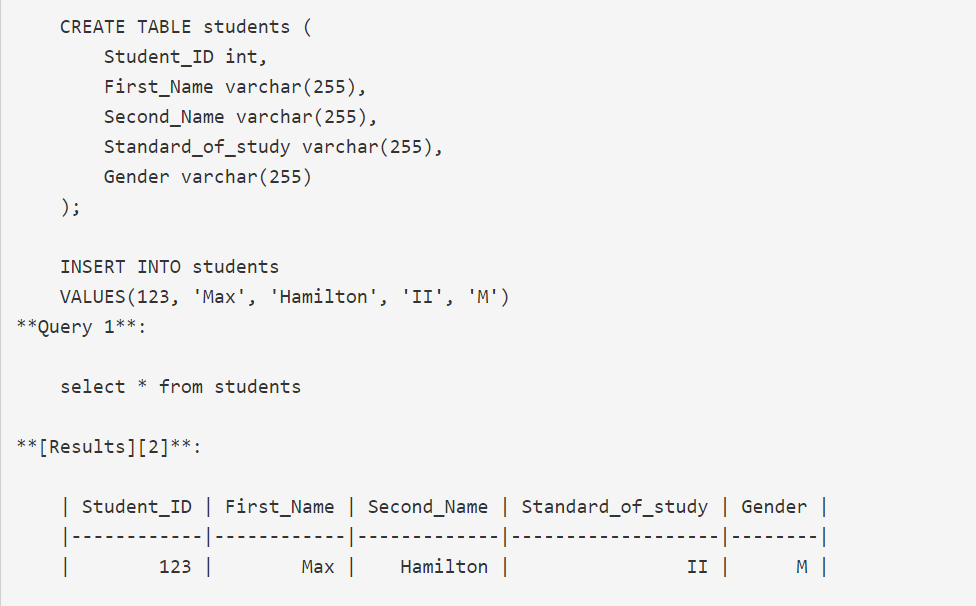
Now that we have created our first table, the next step will be to retrieve this data. This will be a straightforward process of querying the table with its name using the SELECT and FROM commands.



We can retrieve data by using the SELECT and FROM commands

When a table consists of multiple rows, we can use the **WHERE** command along with **SELECT** to retrieve specific rows. The WHERE clause is generally used with a combination of various conditions for filtering the data while retrieving it from either created tables or retrieving tables from databases.

To look at the entire flow of the code at once,



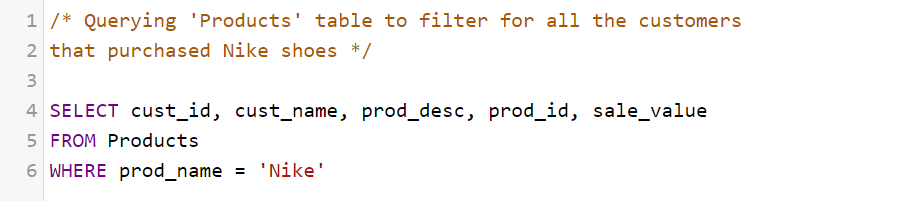
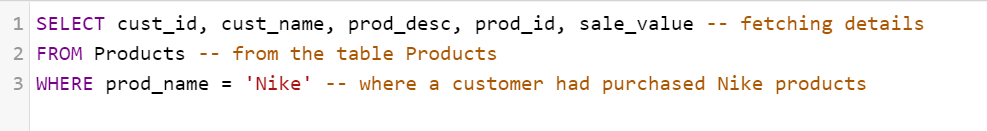
We hope this section has intrigued your interest in seeing how simple it is to start with learning SQL. We will be talking about the steps that you can undertake to master SQL for DS in the last section of this article, so stick around until then.

# Adding Comments to SQL

We have emphasized how crucial it is to include comments in the SQL queries that you write. It improves on the following aspects:

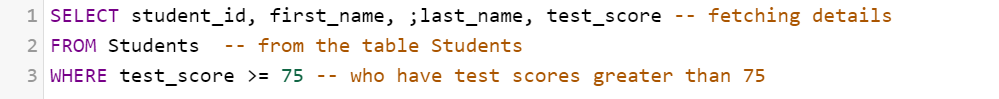
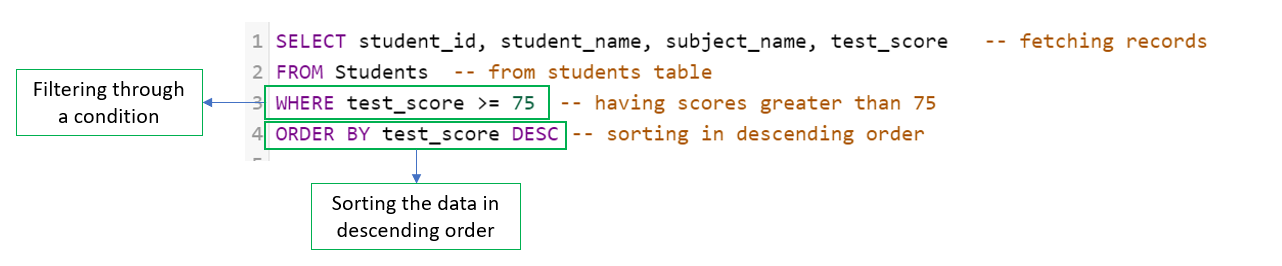
* The readability of the query, as the purpose of the query shall be included with each code chunk created
* The query shall appear much cleaner which makes it more consumable for any other person trying to debug or use it
* Plus, as “*extra marks for good handwriting”* goes, other users will definitely be thanking us for maintaining the standards

Looking at all these pointers, let's get into the practice of making our code cleaner by glancing into the methods to apply for commenting on our SQL queries and codes. We will be looking at two methods to add comments to our code:

* Begin the comment with **‘/\*’**, then proceed with the content of the comment which can span multiple lines as well. The comment can be terminated with **‘\*/’**. For example:   
  
* Another way to add comments to your query is by using ‘--’. The advantage of using this method is that it can be added along the lines of code as we can see in the example here:  
    
    
  Commenting on your code becomes essentially necessary as you start writing more advanced queries. As codes start becoming more complex, adding these blocks will help you a great deal in comprehending your own code.

# Filtering, Sorting and Calculating Data with SQL

As we had mentioned that being a data scientist you will spend almost 80% of your time understanding your data. Here we are going to introduce a few concepts that will help you approach this process.

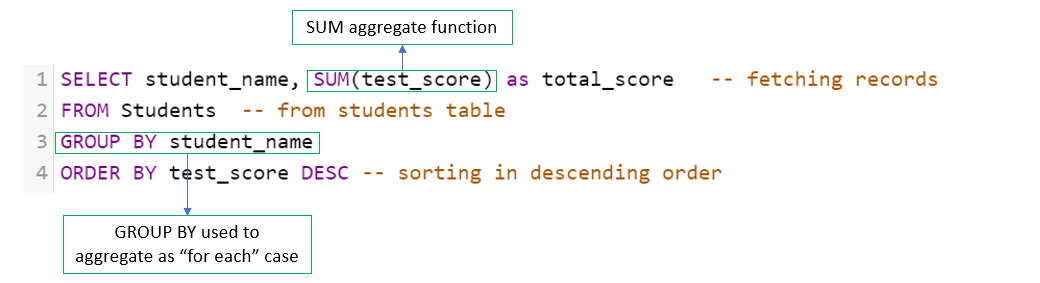
Filtering data and Sorting Data You might have already observed that we have already used the WHERE clause in a few of our examples above. **WHERE** clause within SQL helps us filter our data from the database at a level that you will want to see your data. It basically filters records from a table that fulfil a specific condition specified by you.   
  
For instance, let's say from your **Students** (A table containing student records with their test scores) table you wanted to have a look at the student scores, but only want to focus on those who scored greater than 75. You also want to sort them in descending order to rank them, then your query will look like the following chunk:   
  
  
  


#### Image by [Author](https://www.linkedin.com/in/anishmahapatra/) You can have multiple conditions included in the **WHERE** clause and can various operands to slice your data at a level you are seeking to fetch it.

The **ORDER BY** clause is simply used for sorting the data in either ascending or descending order. The default setting for ORDER BY is sorting the values in ascending order. The ORDER BY clause must be included only after the filtering clause of WHERE as stated by the order of execution in SQL.

## Aggregating/Calculating Data

Another direction we can look into for SQL’s functions is the aggregation section. Using the same example above, you want to aggregate scores for a student across all the subjects. You will be using aggregate functions as shown below:

*Image by* [*Author*](https://www.linkedin.com/in/anishmahapatra/)

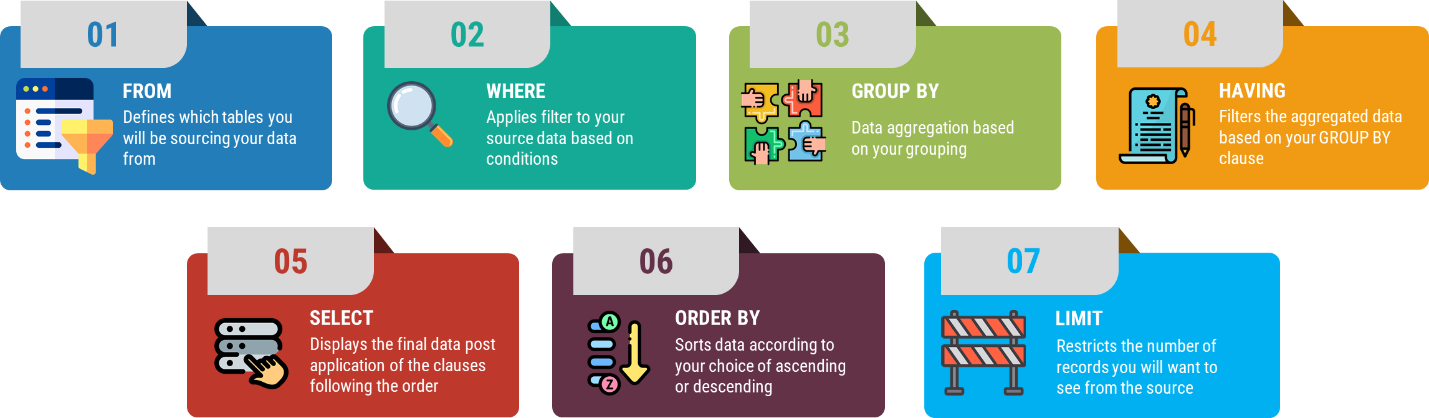
Aggregate functions are used with the **GROUP BY** clause which will create groups of specified columns and apply the aggregation functions to them.

Here, we would strongly recommend you understand the ***order of execution in SQL*** (which we shall introduce in the next section) \*\*\*\*\*\*as well as most of the queries you will be working on will be a combination of many clauses post the SELECT statement.

# 6 SQL Tricks Every Data Scientist Should Know

## Order of execution

Firstly, it gets really confusing with knowing so many clauses that you can use to fetch your data but not surely know where to place which of these. You will be inviting trouble for yourself if you were to mess up the order of execution



#### Image by [Author](https://www.linkedin.com/in/anishmahapatra/)

## Master the “GROUP BY” clause in SQL

One of the most often used clauses in SQL is "GROUP BY". It can be used to aggregate data, and there are numerous ways to make it work effectively for you. The different ways you can think of using it are:

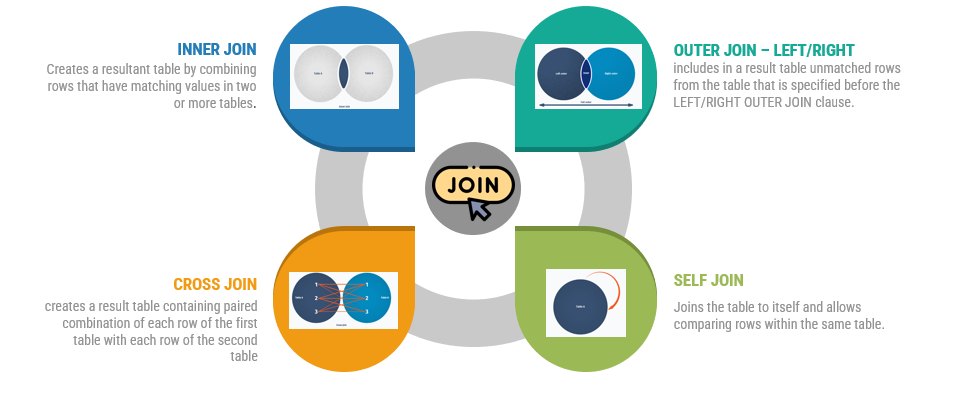
GROUP BY with one or multiple columns

1. Use ORDER BY for sorting your data post application of GROUP BY
2. HAVING clause is specifically used with GROUP BY to filter the aggregated/grouped data
3. These are a few tips you can begin with and dive deeper into using GROUP BY with more complex queries

## Joins

Often you might require to analyze data tables that might contain information in more than one table. It becomes a tedious task to look into both of these tables separately to come up with an analysis. To save you time with this, there exist multiple types of joins.

Joins indicate how the SQL server should use the data from one table to select the rows from another table. The different types of joins that you use are:



#### Image by [Author](https://www.linkedin.com/in/anishmahapatra/)

Mastering Joins in SQL will help you deal with multiple tables at once post the application of joins. Although, one aspect you need to be extra careful about is the concepts of types of keys in SQL which are primary, secondary, foreign, composite, super, and alternate keys. Knowing them will help you in effectively using Join statements.

## Nested queries or Subqueries

A Subquery or a Nested query is a query within another SQL query and is embedded within the WHERE clause. Subqueries come in handy when the result that you are looking for requires more than a single query and each subquery provides a subset of the table involved in the query.

## Stored Procedures

Python user-defined functions or UDFs are quite well known around and used extensively in the Python environment. SQL's equivalent of User Defined Functions is Stored Procedures. A stored procedure is basically a prepared SQL code that can be saved and used multiple times to perform the same action

## Working with Indexes and Null Values

With the help of indexes, we can load the data easily onto the database. Indexes also help us with special lookup tables. So including indexes can add benefits while querying the data.

Null values can cause issues in the future if they are not dealt with in the data preparation step. We might face issues like not being able to use formulas, calculations and other analyses to get accurate results. So, it becomes necessary to understand null values and deal with them by applying business context.

# Steps to Master SQL for Data Science [From Scratch]

Moving on to the final element here, we would like to highlight how you can master SQL specifically for Data Science. Fundamentally, data science is all about understanding data and using it to create impactful insights through analysis and modelling using AI/ML that can drive decisions to bring about changes in behaviour. So where does SQL fit into this equation here?

We know the data science pipeline consists of a crucial step to creating a dataset that will enable us to conduct our analysis, run our ML models, and gather findings and insights. We also call it the ‘**Analytical Dataset**’ or the ADS. There have been multiple occasions where one might have wondered how easier it might have been to utilize SQL to create the analytical dataset and move it into further stages.

With that idea in mind, we capitalized on this article on learning SQL for data science. All the concepts we touched upon in this article will help you learn the notions required to create/handle datasets, query and filter them, add aggregations, group them and also utilize window functions over them to get that ADS ready for further analysis.

Our one favourite open-source website which we found extremely convenient to learn so many concepts of SQL which are relevant to Data Science is [SQLZOO](https://sqlzoo.net/wiki/SQL_Tutorial). This website consists of a step-by-step tutorial on all the basic topics that we have mentioned here.

# Conclusion

In this article, we have gone over in immense detail how you can master SQL for Data Science. I hope you will appreciate the amount of effort that has gone to create the content for you. In case you have any questions, please feel free to reach out. We went over SQL all the way from its origin to how it’s still one of the most popular languages to date.

One of the main reasons why SQL is still so prevalent is its integration with other languages such as Python, Spark, R and Hadoop. Its constant evolving usage across multiple domains is helping SQL stay the #1 choice for most data-related interviews, given its extensive application in the enterprise!

# Author

**Anish Mahapatra**   
*Senior Data Scientist*   
LinkedIn: [https://www.linkedin.com/in/anishmahapatra](about:blank) ([LinkedIn](about:blank))   
Email: anishmahapatra01@gmail.com