Contents

[Prerequisites 1](#_Toc83116104)

[What are Data? 1](#_Toc83116105)

[What is a Database? 1](#_Toc83116106)

[What is a Database Management System (DBMS)? 1](#_Toc83116107)

[Types of DBMS 2](#_Toc83116108)

[What is Database? What is SQL? 2](#_Toc83116109)

[What is SQL? 3](#_Toc83116110)

[What is NoSQL? 3](#_Toc83116111)

[Why it makes sense to learn SQL after NOSQL? 4](#_Toc83116112)

[Reference 7](#_Toc83116113)

# Prerequisites

## What is Data?

In simple words, data can be facts related to any object in consideration.

For example, your name, age, height, weight, etc are some data related to you.

A picture, image, file, pdf etc can also be considered data.

## What is a Database?

Database is a systematic collection of data. Databases support storage and manipulation of data. Databases make data management easy. Let us discuss few examples.

An online telephone directory would definitely use database to store data pertaining to people, phone numbers, other contact details, etc.

Your electricity service provider is obviously using a database to manage billing, client related issues, to handle fault data, etc.

Let us also consider the Facebook. It needs to store, manipulate and present data related to members, their friends, member activities, messages, advertisements and lot more.

## What is a Database Management System (DBMS)?

Database Management System (DBMS) is a collection of programs, which enables its users to access database, manipulate data, reporting / representation of data.

It also helps to control access to the database.

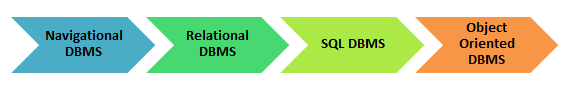
Database Management Systems are not a new concept and as such had been first implemented in 1960s.

Charles Bachmen's Integrated Data Store (IDS) is said to be the first DBMS in history.

With time database technologies evolved a lot while usage and expected functionalities of databases have been increased immensely.

## Types of DBMS

Let us see how the DBMS family got evolved with the time. Following diagram shows the evolution of DBMS categories.



## What is Database? What is SQL?

There are four major types of DBMS. Let's look into them in detail.

**Hierarchical** - this type of DBMS employs the "parent-child" relationship of storing data. This type of DBMS is rarely used nowadays. Its structure is like a tree with nodes representing records and branches representing fields. The windows registry used in Windows XP is an example of a hierarchical database. Configuration settings are stored as tree structures with nodes.

**Network DBMS** - this type of DBMS supports many-to many relations. This usually results in complex database structures. RDM Server is an example of a database management system that implements the network model.

**Relational DBMS** - this type of DBMS defines database relationships in form of tables, also known as relations. Unlike network DBMS, RDBMS does not support many to many relationships. Relational DBMS usually have pre-defined data types that they can support. This is the most popular DBMS type in the market. Examples of relational database management systems include MySQL, Oracle, and Microsoft SQL Server database.

**Object Oriented Relation DBMS** - this type supports storage of new data types. The data to be stored is in form of objects. The objects to be stored in the database have attributes (i.e. gender, ager) and methods that define what to do with the data. PostgreSQL is an example of an object oriented relational DBMS.

## What is SQL?

Structured Query language (SQL) pronounced as "S-Q-L" or sometimes as "See-Quel"is actually the standard language for dealing with Relational Databases.

SQL programming can be effectively used to insert, search, update, and delete database records. That doesn't mean SQL cannot do things beyond that.

In fact, it can do lot of things including, but not limited to, optimizing and maintenance of databases.

Relational databases like MySQL Database, Oracle, Ms SQL server, Sybase, etc uses SQL! How to use sql syntaxes?

SQL syntaxes used in these databases are almost similar, except the fact that some are using few different syntaxes and even proprietary SQL syntaxes.

**SQL Example**

SELECT \* FROM Members WHERE Age > 30

## What is NoSQL?

NoSQL is a category of Database Management Systems. Its main characteristic is its non-adherence to Relational Database Concepts. NOSQL means "Not only SQL".

Concept of NoSQL databases grew with internet giants such as Google, Facebook, and Amazon etc who deal with gigantic volumes of data.

When you use relational database for massive volumes of data, the system starts getting slow in terms of response time.

To overcome this, we could of course "scale up" our systems by upgrading our existing hardware.

The alternative to the above problem would be to distribute our database load on multiple hosts as the load increases.

This is known as "scaling out".

NOSQL database are non-relational databases that scale out better than relational databases and are designed with web applications in mind.

They do not use SQL to query the data and do not follow strict schemas like relational models. With NoSQL, ACID (Atomicity, Consistency, Isolation, and Durability) features are not guaranteed always

## Why it makes sense to learn SQL after NOSQL?

With the advantages of NOSQL databases outlined above that scale out better than relational models, you might be thinking why one would still want to learn about SQL database?

Well, NOSQL databases are somewhat highly specialized systems and have their special usage and limitations. NOSQL suit more for those who handles huge volumes of data. The vast majority use relational databases and associated tools.

Relational databases have the following advantages over NOSQL databases;

SQL (relational) databases have a mature data storage and management model. This is crucial for enterprise users.

SQL databases support the notion of views, which allow users to only see data that they are, authorized to view. The data that they are not authorized to see is kept hidden from them.

SQL databases support stored procedure sql, which allow database developers to implement part of the business logic into the database.

SQL databases have better security models compared to NoSQL databases.

The world has not deviated from use of relational databases. There is growing a demand for professionals who can handle relational databases. Thus, learning databases and SQL still holds merit.

## Topics

**Basics:**

* [Structured Query Language](https://www.geeksforgeeks.org/structured-query-language/)
* [SQL | Datatypes](https://www.geeksforgeeks.org/sql-datatypes/)
* [SQL | DDL, DML, TCL and DCL](https://www.geeksforgeeks.org/sql-ddl-dml-tcl-dcl/)
* [SQL | TRANSACTIONS](https://www.geeksforgeeks.org/sql-transactions/)
* [SQL | VIEWS](https://www.geeksforgeeks.org/sql-views/)
* [SQL | Comments](https://www.geeksforgeeks.org/sql-comments/)
* [SQL | Constraints](https://www.geeksforgeeks.org/sql-constraints/)
* [SQL | Creating Roles](https://www.geeksforgeeks.org/sql-creating-roles/)
* [SQL | Indexes](https://www.geeksforgeeks.org/sql-indexes/)
* [SQL | SEQUENCES](https://www.geeksforgeeks.org/sql-sequences/)
* [SQL | Query Processing](https://www.geeksforgeeks.org/sql-query-processing/)
* [CTE in SQL](https://www.geeksforgeeks.org/cte-in-sql/)
* [SQL Trigger | Student Database](https://www.geeksforgeeks.org/sql-trigger-student-database/)

**Clauses / Operators:**

* [SQL | WITH clause](https://www.geeksforgeeks.org/sql-with-clause/)
* [SQL | With Ties Clause](https://www.geeksforgeeks.org/sql-ties-clause/)
* [SQL | Arithmetic Operators](https://www.geeksforgeeks.org/sql-arithmetic-operators/)
* [SQL | Wildcard operators](https://www.geeksforgeeks.org/sql-wildcard-operators/)
* [SQL | Intersect & Except clause](https://www.geeksforgeeks.org/sql-intersect-except-clause/)
* [SQL | USING Clause](https://www.geeksforgeeks.org/sql-using-clause/)
* [SQL | MERGE Statement](https://www.geeksforgeeks.org/sql-merge-statement/)
* [MERGE Statement in SQL Explained](https://www.geeksforgeeks.org/merge-statement-sql-explained/)
* [SQL | DDL, DML, DCL and TCL Commands](https://www.geeksforgeeks.org/sql-ddl-dml-dcl-tcl-commands/)
* [SQL | CREATE DOMAIN](https://www.geeksforgeeks.org/sql-create-domain/)
* [SQL | DESCRIBE Statement](https://www.geeksforgeeks.org/sql-describe-statement/)
* [SQL | Case Statement](https://www.geeksforgeeks.org/sql-case-statement/)
* [SQL | UNIQUE Constraint](https://www.geeksforgeeks.org/sql-unique-constraint/)
* [SQL | Create Table Extension](https://www.geeksforgeeks.org/sql-create-table-extension/)
* [SQL | ALTER (RENAME)](https://www.geeksforgeeks.org/sql-alter-rename/)
* [SQL | ALTER (ADD, DROP, MODIFY)](https://www.geeksforgeeks.org/sql-alter-add-drop-modify/)
* [SQL | LIMIT Clause](https://www.geeksforgeeks.org/sql-limit-clause/)
* [SQL | INSERT IGNORE Statement](https://www.geeksforgeeks.org/sql-insert-ignore-statement/)
* [SQL | LIKE](https://www.geeksforgeeks.org/sql-like/)
* [SQL | SOME](https://www.geeksforgeeks.org/sql-some/)
* [SQL | OFFSET-FETCH Clause](https://www.geeksforgeeks.org/sql-offset-fetch-clause/)
* [SQL | Except Clause](https://www.geeksforgeeks.org/sql-except-clause/)
* [Combining aggregate and non-aggregate values in SQL using Joins and Over clause](https://www.geeksforgeeks.org/combining-aggregate-and-non-aggregate-values-in-sql-using-joins-and-over-clause/)
* [SQL | ALL and ANY](https://www.geeksforgeeks.org/sql-all-and-any/)
* [SQL | EXISTS](https://www.geeksforgeeks.org/sql-exists/)
* [SQL | GROUP BY](https://www.geeksforgeeks.org/sql-group-by/)
* [SQL | Union Clause](https://www.geeksforgeeks.org/sql-union-clause/)
* [SQL | Aliases](https://www.geeksforgeeks.org/sql-aliases/)
* [SQL | ORDER BY](https://www.geeksforgeeks.org/sql-order-by/)
* [SQL | SELECT TOP Clause](https://www.geeksforgeeks.org/sql-select-top-clause/)
* [SQL | UPDATE Statement](https://www.geeksforgeeks.org/sql-update-statement/)
* [SQL | DELETE Statement](https://www.geeksforgeeks.org/sql-delete-statement/)
* [SQL | INSERT INTO Statement](https://www.geeksforgeeks.org/sql-insert-statement/)
* [SQL | AND and OR operators](https://www.geeksforgeeks.org/sql-and-and-or-operators/)
* [SQL | WHERE Clause](https://www.geeksforgeeks.org/sql-where-clause/)
* [SQL | Distinct Clause](https://www.geeksforgeeks.org/sql-distinct-clause/)
* [SQL | SELECT Query](https://www.geeksforgeeks.org/sql-select-query/)
* [SQL | DROP, TRUNCATE](https://www.geeksforgeeks.org/sql-drop-truncate/)
* [SQL | CREATE](https://www.geeksforgeeks.org/sql-create/)
* [SQL | Join (Cartesian Join & Self Join)](https://www.geeksforgeeks.org/sql-join-cartesian-join-self-join/)
* [SQL | Alternative Quote Operator](https://www.geeksforgeeks.org/sql-alternative-quote-operator/)
* [SQL | Concatenation Operator](https://www.geeksforgeeks.org/sql-concatenation-operator/)
* [SQL | MINUS Operator](https://www.geeksforgeeks.org/sql-minus-operator/)
* [SQL | DIVISION](https://www.geeksforgeeks.org/sql-division/)
* [SQL | NOT Operator](https://www.geeksforgeeks.org/sql-not-operator/)
* [SQL | BETWEEN & IN Operator](https://www.geeksforgeeks.org/sql-between-in-operator/)
* [SQL | Join (Inner, Left, Right and Full Joins)](https://www.geeksforgeeks.org/sql-join-set-1-inner-left-right-and-full-joins/)
* [SQL | CHECK Constraint](https://www.geeksforgeeks.org/sql-check-constraint/)

**Functions:**

* [SQL | Mathematical functions (SQRT, PI, SQUARE, ROUND, CEILING & FLOOR)](https://www.geeksforgeeks.org/sql-server-mathematical-functions-sqrt-pi-square-round-ceiling-floor/)
* [SQL | Conversion Function](https://www.geeksforgeeks.org/sql-conversion-function/)
* [SQL general functions | NVL, NVL2, DECODE, COALESCE, NULLIF, LNNVL and NANVL](https://www.geeksforgeeks.org/sql-general-functions-nvl-nvl2-decode-coalesce-nullif-lnnvl-nanvl/)
* [SQL | Conditional Expressions](https://www.geeksforgeeks.org/sql-conditional-expressions/)
* [SQL | Character Functions with Examples](https://www.geeksforgeeks.org/sql-character-functions-examples/)
* [SQL | Date Functions (Set-1)](https://www.geeksforgeeks.org/sql-date-functions-set-1/)
* [SQL | Date Functions (Set-2)](https://www.geeksforgeeks.org/sql-date-functions-set-2/)
* [SQL | LISTAGG](https://www.geeksforgeeks.org/sql-listagg/)
* [SQL | Aggregate functions](https://www.geeksforgeeks.org/database-management-system-aggregate-functions/)
* [SQL | Functions (Aggregate and Scalar Functions)](https://www.geeksforgeeks.org/sql-functions-aggregate-scalar-functions/)
* [SQL | Date functions](https://www.geeksforgeeks.org/sql-date-functions/)
* [SQL | NULL](https://www.geeksforgeeks.org/sql-null-functions/)
* [SQL | Numeric Functions](https://www.geeksforgeeks.org/sql-numeric-functions/)
* [SQL | String functions](https://www.geeksforgeeks.org/sql-string-functions/)
* [SQL | Advanced Functions](https://www.geeksforgeeks.org/sql-advanced-functions/)

**Queries:**

* [SQL | Joining three or more tables](https://www.geeksforgeeks.org/joining-three-tables-sql/)
* [SQL | How to Get the names of the table](https://www.geeksforgeeks.org/get-names-table-sql/)
* [SQL | Sub queries in From Clause](https://www.geeksforgeeks.org/sql-sub-queries-clause/)
* [SQL | Correlated Subqueries](https://www.geeksforgeeks.org/sql-correlated-subqueries/)
* [SQL | Top-N Queries](https://www.geeksforgeeks.org/sql-top-n-queries/)
* [SQL | SUB Queries](https://www.geeksforgeeks.org/sql-sub-queries/)
* SQL | SP
* SQL | Function
* SQL | Pivot
* CLR

**Advance:**

* XML Read
* Json Read
* Grouping sets
* SQL profiling

**SQL-Injection:**

* [SQL Injection](https://www.geeksforgeeks.org/sql-injection-2/)
* [How to use SQLMAP to test a website for SQL Injection vulnerability](https://www.geeksforgeeks.org/use-sqlmap-test-website-sql-injection-vulnerability/)
* [Mitigation of SQL Injection Attack using Prepared Statements (Parameterized Queries)](https://www.geeksforgeeks.org/mitigation-sql-injection-attack-using-prepared-statements-parameterized-queries/)
* [Basic SQL Injection and Mitigation with Example](https://www.geeksforgeeks.org/basic-sql-injection-mitigation-example/)

# Reference

**URL**: <https://www.geeksforgeeks.org/sql-tutorial/?ref=ghm>