SQL and PostgreSQL both have a similar syntax.

PostgreSQL have certain extra functionalities that are not there in SQL which are as follows:

1. Support many advanced types such as array, hstore, and user-defined type.
2. Supports IP Address Data type.
3. Supports partial and bitmap indexes.
4. Wide range of options to setup triggers.
5. Supports CASCADE operations.
6. Supports extensions and custom modules.
7. Supports unicode characters.
8. Supports JSON by default.
9. Completely open source.
10. Works on almost every OS.

So it is the best in the open source environment so far.

Despite the fact that PostgreSQL and Microsoft SQL Server come with different database editions, which are usually dependent on the developer's specifications, these two softwares are widely used in relational database management systems. They are also compatible with numerous small and large enterprise applications.

Relational Database Management System allows users to work with back-end data using a variety of commands such as creates, updates, deletes, and reads. This is because when developers build software projects, they often use an RDBMS. They can also choose from a variety of database applications based on their requirements. A commercial or open-source database may also be chosen by the programmer. Even so, most developers opt for a commercial database because it offers more sophisticated functionality than free software.

Difference between MS SQL Server and PostgreSQL:-

S.NO.	MS SQL SERVER	POSTGRESQL
1.	Developed by Microsoft Corporation and initially released on April 24, 1989	Developed by PostgreSQL Global Development Group on 1989.
2.	MS SQL server is written in C++ language.	PostgreSQL is written in C language.
3.	It is a	It is widely used

S.NO.	MS SQL SERVER	POSTGRESQL
	Microsoft relational DBMS.	open source RDBMS.
4.	The primary database model for MS SQL Server is Relational DBMS.	The primary database model for PostgreSQL is also Relational DBMS.
5.	It also has two Secondary database models – Document store and Graph DBMS.	It has Document store as Secondary database models.
6.	The license for MS SQL Server is Commercial.	The license for is Open Source.
7.	Server operating systems for MS SQL Server are Linux and Windows.	Server operating systems for PostgreSQL are FreeBSD, HP-UX, Linux, NetBSD, OpenBSD, OS X, Solaris, Unix and Windows.
8.	In MS SQL Server, partitioning methods are Horizontal partitioning and Sharding through federation.	In PostgreSQL, partitioning can be done by range, list and hash.
9.	It support	It support only one

S.NO.	MS SQL SERVER	POSTGRESQL
	replication but in depends on the SQL-Server Edition.	replication methods – Master-master replication.
10.	It supports in-memory capabilities.	It does not supports in-memory capabilities.

Which Should You Choose for a New Project?

Choosing the DBMS you will use for a new project is a very important and difficult decision. Two popular choices are MS SQL Server and PostgreSQL. To help you decide which would be best for you, I'll compare their features, list pros and cons, and give you some examples.

Your selection of database management system (DBMS) depends on the type of business or project you're implementing. First, you need to know:

1. The type of application(s) you are planning to develop.
2. How you want your data to be available, usable, and protected.
3. Your backup and recovery policy.
4. Your data center or hosting options.
5. Your budget for different phases, including licensing and ongoing maintenance and upgrade costs during your project's lifetime.

Availability

High availability has always been a priority requirement for a database. High-availability features include failover deployment, near-instantaneous failover or automatic failback, failover to cloud, and notification and alerting options.

In **PostgreSQL**, high availability through load balancing and replication features make it a very reliable database. Its architecture enables multiple database servers to work together, which allows a standby server to take over instantaneously if the primary server fails. It also has other high-availability solutions, such as data partitioning, shared disk failover, and write-ahead log shipping.

The *EDB Postgres Automatic Failover Manager* monitors and identifies the causes of database failures. It also automatically performs load-balancing operations as well as messaging and alerting database administration and management. Postgres' synchronization solution is another robust tool that ensures the system's availability, although there is a tradeoff between its functionality and overall database performance.

In **MS SQL Server**, the Enterprise Editions of MS SQL Server 2017 and up have introduced two different architectures for database availability. Their *Always ON Availability Groups* solution is designed for high availability; the *Read Scale Availability Group* architecture handles read-only workload balancing but not high availability. A cluster manager is required for *Always ON Availability Groups*. It is in the Windows Server Failover Cluster (WSFC) in Windows and in the Pacemaker in Linux. There is no cluster manager for the Read Scale Availability Group tool.

Data Security

Data security is one of the most important requirements of database implementation. The scale of online criminal activities – including data theft, piracy, and hacking – has significantly grown with the rapid advancement of online technologies as well as data highways. Hence, any DBMS you implement must ensure data security. Data security can be further categorized as follows:

- Server-level data protection.
- User management and authentication, access protection, and encryption.

Server level data protection

PostgreSQL's server-level advanced authentication methods include LDAP (Lightweight Directory Access Protocol) and PAM (Pluggable Authentication Module), which provide protection from attacks. These security mechanisms potentially reduce the attack surface of the PostgreSQL database servers. Other server-level security enhancements to PostgreSQL include PostgreSQL server listen address, host-based authentication, and certificate authentication.

In **MS SQL Server**, there are two server-level security enhancement features: *Windows Authentication Mode* and *Mixed Mode*. The security model of MS SQL Server is a tight integration between the Windows Authentication mode of Windows Server and the database. Windows Authentication Mode performs better in the following scenarios:

- When there is a domain controller.
- With SQL Server Express or LocalDB database instances.
- When both the application and the database are in the same machine environment.

Mixed Mode includes an authentication process by both Windows Server and MS SQL Server. Here, the database deploys the mechanisms of Windows Password Policy. The designed complexity of Windows Password Policy is aimed at preventing hacking attacks.

User Management and Authentication, Access Protection, and Encryption

These are the deep-end security enhancement features of the DBMS. Different databases have different types and levels of data encryption methods, which provide protection against data theft.

On the other hand, the most common requirements of today's robust business applications include concurrent access by multiple users, integration and synchronization with third party applications, role-based authentication policies and profiles, etc. In addition to having built-in access protection, user management and encryption technology in today's databases also includes the ability to integrate with other powerful access protection and encryption tools.

PostgreSQL provides data encryption and allows you to use SSL (Secure Sockets Layer) certificates when your data is travelling through the web or public network highways. It also allows you to implement Client Certificate Authentication tools as an option. In addition, you can use cryptogenic functions to store encrypted data in PostgreSQL. These cryptogenic functions support both symmetric-key and public-key encryptions.

User management (i.e. assigning roles and privileges to individual users) is another common data security feature. In PostgreSQL, you can implement this using role-based access control, which includes user-level privileges as role assignments, table-level privileges via roles, and role inheritance.

For example, there may be many user accounts in your PostgreSQL database server (i.e. application accounts, end user accounts, admin accounts, developer accounts, etc.). You do not want operators to access tables meant for managers; most businesses make sure that only designated personnel access the end users' data or confidential/sensitive business information. PostgreSQL allows you to create role-based profiles with a certain set of privileges for each group of users. Thus, you can manage both user groups and individual user roles based on a package of privileges, which will help you ensure access protection. In addition, the auditing option allows you to review users' and groups' data access activities in your database, which provides a layer of extra security.

In **MS SQL Server**, the available data encryption features include Transparent Data Encryption (TDE), Always Encrypted, and column-level encryption. TDE uses the Advanced Encryption Standard (AES) algorithm for encrypting physical files, which include both data and log files. The Always Encrypted feature allows you to encrypt

certain columns in both states, at rest and in motion (i.e. the data remains encrypted in memory as well).

SQL Server's adopted security features include simplified user management via user groups and roles. This includes:

- Explicit data or resource permissions granted directly to the user account.
- Inherited permissions, as part of a user group.
- Permissions inherited from a parent resource.

Auditing and monitoring activities in MS SQL Server lets you identify concurrency issues, malicious long-running queries, and regular workload metrics.

Performance

Different database engines have different levels of performance tuning options in their DBA dashboards. These are mainly based on setting ratios of different functionality parameters, such as ratios of data reads to writes, etc. Good performance does not always mean that the database is fast; rather, the balance of the database engine, in terms of utilizing different physical resources (i.e. memory, processors, and storage servers) is important. Although there is built-in auto-performance tuning available in most database engines, customizable performance optimization and monitoring features are the savior tools that keep you from disasters when there is huge data traffic and the database becomes gigantic.

PostgreSQL has many tools and parameters for monitoring and optimizing the database performance. It has a proven high performance rating in both online transaction processing (OLTP) and online analytical processing (OLAP). The multi-version concurrency control (MVCC) feature for the simultaneous processing of multiple transactions with almost no deadlock is another advanced performance milestone achieved by PostgreSQL.

There are some performance optimization limitations in the **MS SQL Server** Standard Edition, including indexing and memory partitioning, etc. However, these limitations are removed in its Enterprise Edition. SQL Server's In-Memory OLTP feature ensures high performance by using in-memory data tables instead of writing directly to the disk. The analytical and transaction processing speed in MS SQL Server is also good.

Data Consistency

The "safety net" of your database engine are its backup and restore features:

- Backup consistency.
- Backup speed and processing times.
- Restore times (if something goes wrong).

These are important because when your business is running, any unannounced downtime and/or data loss is the worst possible scenario.

PostgreSQL has built-in logical backup utilities, such as pg_dump and pg_dumpall. Also, there are many trusted third-party data consistency tools available for PostgreSQL, such as Amanda, Bacula, Barman, Handy Backup, Iperius Backup, NetVault Backup, pg_probackup, pgBackRest, Simpana, Spectrum Protect, and Veritas NetBackup for PostgreSQL Agent.

MS SQL Server provides three online backup strategies: simple, full, and bulk-logged recovery models. The most recommended tool is the full recovery model, in which no data loss is acceptable. In this model, database recovery at any point in time is possible. Regular database and transaction log backup is mandatory for the SQL Server's full recovery model. The bulk-logged recovery model is used as a temporary solution, and the simple recovery model works well if your database is rarely updated.

PostgreSQL vs. MS SQL Server: Application Related Factors

Scalability

This very important factor determines how efficiently your chosen DBMS can scale to meet your requirements. In other words, the capacity of the system needs to grow as your business grows and more data comes in. It is essential to estimate how easily a DBMS can manage a large scale of data, even if the database grows very quickly after the project is operational.

There are two types of partitioning options and several indexing options available in **PostgreSQL** that improve query performance and other data operations. These indexes and partitions are segregated in different disks' storage, which enhances table scalability.

The improved scalability in **MS SQL Server** is a development of the sharding design pattern of data storage. Here, data storage is divided into a set of horizontal partitions (according to database sharding), as shown below:

SQL Server also allows you to run multiple concurrent threads in memory-optimized tables in addition to the other scalability features, including dynamic management views and multithreaded recovery and merge operations.

Usability

Your database has different user groups, including non-technical data consumers as well as your technical team (database administrators, application developers, and integrators). It is important to know that the easier it is for any of the aforementioned user groups to work with the database, the lower the ongoing cost will be.

With a full stack of RDBMS features and data-handling capabilities, **PostgreSQL** is both easy to use and an advanced object-relational database management system. It uses Structured Query Language (SQL) in addition to a procedural language called PL/SQL. Check out our track on SQL FUNDAMENTALS IN POSTGRESQL to learn how to talk to a PostgreSQL database.

Reporting Tools and Other GUI Interfaces

Depending on the type of your project or application, it is sometimes essential to find out how smart your chosen database's reporting tools are. For example, GUI interfaces are not required for a data warehousing project, but database reporting tools are particularly useful for e-commerce applications and ERP systems. In addition to the reports available in the business application, a reporting tool and other GUI options help business strategists quickly check market changes and data trends.

PostgreSQL has fewer GUI options than MS SQL Server because it is mostly based on the Linux and Unix operating systems and on command platform consoles. The only GUI solution available is EDB Postgres Enterprise Manager. You can intelligently monitor, tune, and manage large-scale PostgreSQL installations from a single GUI console. There are some third party open-source reporting tools available for PostgreSQL, such as Reportizer or dbForge Studio.

Since the first version of **MS SQL Server**, it has included a very intelligent and interactive built-in database management application as well as rich GUI-based reporting tools. From installation up to any extent, the MS SQL Server database is well-equipped in terms of GUI interfaces.

Setup and Installation

Both the PostgreSQL and MS SQL Server databases are easily installable by anyone, so you can start your adventure with the DBMS of your choice in as little as 5 minutes. Check out our article on HOW TO INSTALL POSTGRESQL ON WINDOWS 10 IN 5 MINUTES; if you want to start with MS SQL Server, here is an article on MICROSOFT SQL SERVER 2017 INSTALLATION STEP BY STEP.

Whatever DBMS you choose, you will see how easy it is to get started!

PostgreSQL vs. MS SQL Server: Business Related Factors

Development, Deployment, Maintenance and Upgrade Costs

Licensing cost is one of the essential factors for choosing a database engine. In addition, support, development, and integration costs are all business expenses. You

must also be aware of the upgrade cost and availability of professionals and support within your reach. All these factors have a direct impact on your business performance.

PostgreSQL is an open-source RDBMS. There is no licensing cost for owning and upgrading it. The versions and releases of PostgreSQL database are the same as any vendor-owned DBMS, that is:

- A large group of developers contribute to the development of each version.
- Regular release cycles, with bug fixing and security patches, are maintained.
- Versions have an End of Life (EOL).
- Support is provided until EOL, although in this case it's provided by the Postgres community. For example, PostgreSQL 9.4 reached its EOL stage in February 2020; there is no support available for that version anymore.

There is no licensing and upgrading cost for PostgreSQL – except the operational costs that include the salaries of your in-house database administrators and developers. This operational cost is similar to any other major DBMS.

MS SQL Server has maintained a core-based licensing cost since 2012. The licensing costs of MS SQL Server 2019 version are:

- SQL Server Enterprise Edition: $7,128 per core.
- SQL Server Standard Edition: $1,859 per core.
- SQL Server Standard Edition Server Licensing: $931, plus $209 per named user Client Access License (CAL).

In addition, if the version of the DBMS is approaching EOL, SQL Server does not provide a free upgrade to a newer version. Upgrade and migration to newer versions are at the customers' cost. Operational costs (such as DBA, developer, and manager salaries) are similar to that of any other standard DBMS in the market.

Also, there is a free version of MS SQL Server available to students and application developers.

System Infrastructure

When choosing your database engine, one of the primary decisions you have to make is where your system is going to be located /hosted. Consider:

- What are the costs if you host the data center in-house?
- What data center costs are involved for your chosen database?
- What other infrastructure options can provide smooth, high-performance operations?

PostgreSQL can be installed and managed in a wide range of operating systems and environments, including FreeBSD, HP-UX, Linux, NetBSD, OpenBSD, OS X, Solaris,

Unix, and Windows servers. There is fully managed PostgreSQL hosting on AWS, Azure, and DigitalOcean, with high availability and SSH access on the multi-cloud DBaaS. PostgreSQL also offers a lot of flexibility if you want to have it hosted and operated from your privately-owned data center. The hardware expenses depend on your requirements.

There is a cloud version of MS SQL Server available in Azure. It is known as Azure SQL Database, but it is actually the latest version of SQL Server. SQL Server is also available in the Amazon Web Services (AWS) platform, and you can install it on Windows and Linux servers as well. The hardware and data center costs depend on your requirements. There are many high-security, high-performance, heavy-duty shared data centers available for this DBMS. Owning a private data center has similar hardware, software, services, networking, and labor costs to other RDBMSs.

Support, Service, and Upgrades

In addition to evaluating support and upgrade costs, before you choose a database you should be aware of:

- How support is provided by the database vendor.
- How upgrades will be integrated.
- How the data migration policy works.

All these factors will contribute to your business growth and performance.

There is no cost involved for **PostgreSQL**'s community-based support or its upgrades to newer versions. There might be hardware/data center upgrade and data migration costs when you upgrade versions; plus, the ongoing operational costs are similar to that of any other standard DBMS.

Anyone can have their doubts about the service levels of open-source software; however, the great thing about Postgres is that it continues to grow *because* of the contributions made by community developers. Check out WHICH COMPANIES USE POSTGRESQL as their primary DBMS; it's quite enlightening.

In the case of **MS SQL Server**, support costs depend on the terms and conditions of the licensing agreement. Upgrading to a newer version is at the customer's own expense. Data center upgrades, data migration costs, and other operational costs are similar to that of any other standard DBMS.

Example: Creating an Online Store in PostgreSQL and SQL Server

For creating an e-commerce project, you can use web services (such as Amazon Web Service) to create and connect to a MS SQL Server or PostgreSQL database (e.g. by

using Amazon RDS). You can also download PostgreSQL or the free version of SQL Server and create your own e-commerce database.

You may notice that there are primary key and foreign key relationships among the tables (marked in colors). Here is the SQL code that would create tables for each DBMS:

In PostgreSQL:

```
CREATE TABLE order_details (
    order_detail_id  INT NOT NULL,
    order_id         INT NOT NULL,
    product_id       INT NOT NULL,
    quantity         INT NOT NULL,
    total_amount     INT NOT NULL,
    PRIMARY KEY (order_detail_id),
    FOREIGN KEY (order_id)   REFERENCES order (order_id),
    FOREIGN KEY (product_id) REFERENCES product
(product_id)
);
```

Notice the two foreign keys that link to other tables (the order table and product table, respectively).

In MS SQL Server:

```
CREATE TABLE ecommerce.order_details (
    ord_detail_id  INT PRIMARY KEY IDENTITY (1, 1),
    order_id       INT NOT NULL,
    product_id     INT NOT NULL,
    quantity       INT,
    total_amount   INT,
    FOREIGN KEY (order_id)   REFERENCES ecommerce.order(order_id)
    FOREIGN KEY (product_id) REFERENCES ecommerce.product(product_id)
);
```

MS SQL Server: Pros and Cons

The GUI interface and user-friendly installation and development environment are among the top selling points for MS SQL Server. It has many advanced options, such as compression, partitioning, and optimized storage. Its data management – requiring minimal troubleshooting and with data recovery support in case of data corruption – are very useful.

On the downside, the licensing and support costs are discouraging for the e-commerce users. The advanced features tend to be very expensive. Finally, the rich-GUI client and database management applications are upgraded in every new version; this sometimes requires users to upgrade their hardware in addition to moving to the new version of SQL Server.

SQL Server or PostgreSQL – Which Should You Choose?

Both the PostgreSQL and MS SQL Server databases are among the most popular databases on the market. Hence, I'd say both of them are worth exploring.

One could say that PostgreSQL is for Linux enthusiasts and MS SQL Server is for Windows enthusiasts. However, nowadays that is not valid; both these DBMSs can be installed in any major operating system.

On one hand, MS SQL Server has a more developed GUI; on the other hand, PostgreSQL is open source. Both have their pros and cons, but the final decision is fully dependent on your particular project and requirements.