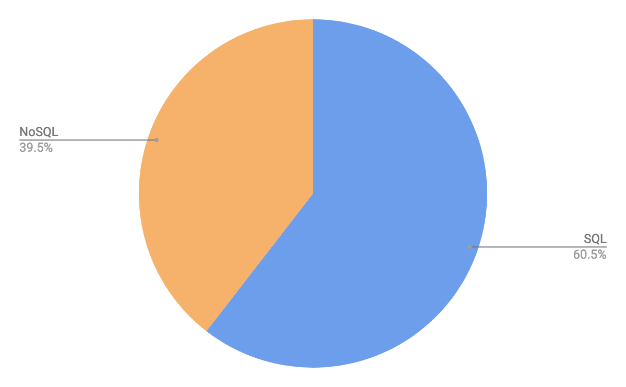
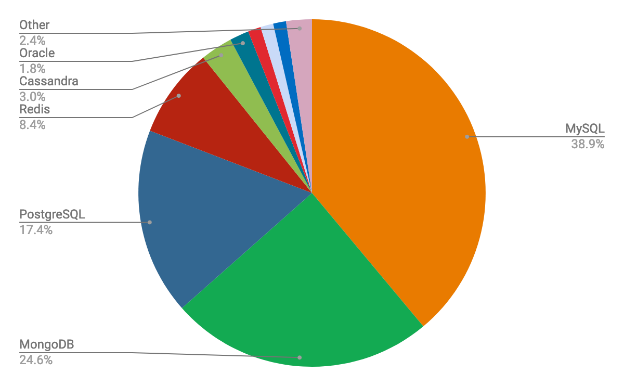
Databases Comparison Task

# Introduction

SQL vs. NoSQL databases. A survey at [DeveloperWeek](https://www.developerweek.com/) revealed that the most popular databases in 2019 were MySQL, MongoDB, PostgreSQL and Redis. However, as we´re in an educational background I´ll just talk about open-source, free projects. The following graphs shows the market share SQL vs. NoSQL and the most used ones:

There are a whole world of databases: ElasticSeach, Hive for NoSQL, Couchbase for SQL, and neo4j or AllegroGraph for Graphs. Among many others, these are experimenting a rise in popularity in the last years. Despite this comparisons and rankings, 44.3% projects use multi-database systems to support their products, and 75.6% use SQL + NoSQL in conjunction (Source 1). These powerful database management systems can complement each other and fill the gaps in terms of data needs.

# Parameters to compare

## Transaction

Transactions allow to bundle multiple database updates, changes… into a single atomic operation and also give you the ability to undo an operation. But this doesn´t come for free, you need to set it up ahead of time.

* Database state: each operation alters the state of the database, moving it forward in time.
* BEGIN; COMMIT; ROLLBACK;

### ACID

* **A**tomic 🡪 all changes to the data must be performed successfully or not at all. You never get half of an operation.
  + Reasons why operations may not work: one or more constraints violated, datatype mismatch, syntax error.
* **C**onsistent 🡪 Data must be in a consistent state (only valid data) before and after the transaction.
* **I**solated: no other process can change the data while the transactions is running.
* **D**urable: the changes made by a transaction must persist.

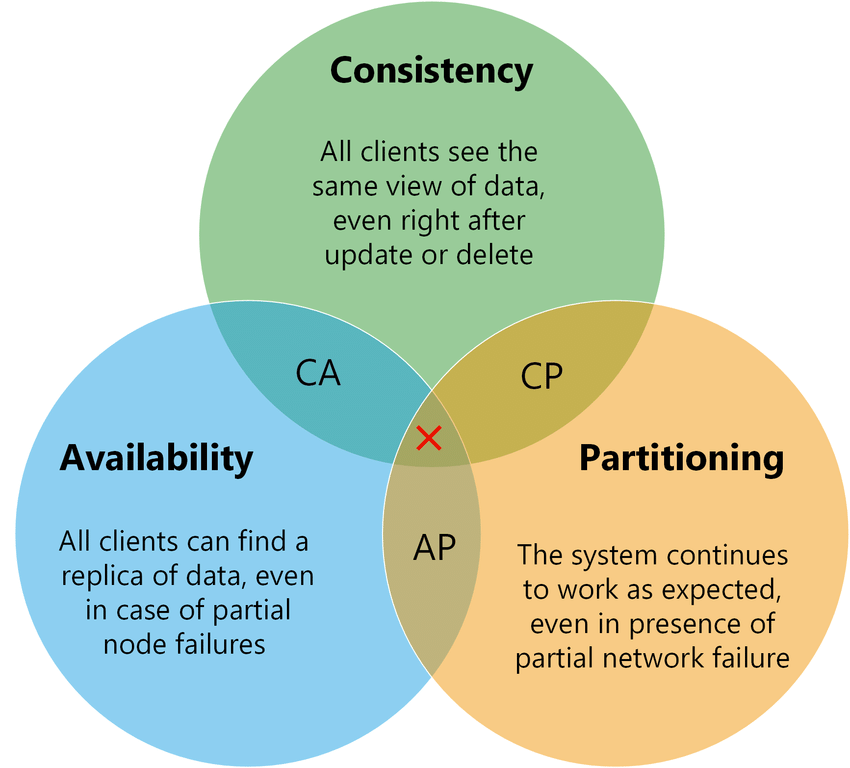
(Source 3)

### Concurrency

Concurrency refers to the simultaneous access to data, which the database must orchestrate carefully to avoid inconsistencies and the measure of how much work the database is doing at a point in time.

## BASE

The CAP theorem states the following: you must choose two from the following diagram:



**Base: B**asically **A**vailable, **S**oft state, **E**ventual consistency

## Scalability

**Vertical scaling:** Updating the characteristics of the server of your database. You must be careful, since if you continue expanding the database, you will find yourself with a bottleneck, the expansion of the characteristics of your server can be finite both in hardware and in budget.

**Horizontal scaling:** Create master / slave environments. For example, you can balance requests by sending the data insert to the main "master" server and the data reads to the slave server

Imagen que contiene taza, alimentos, cuarto

Descripción generada automáticamente

# PostgreSQL

**MySQL-> MariaDB:**

MariaDB is a community-developed, commercially supported fork of the MySQL relational database management system (RDBMS), intended to remain free and open-source software under the GNU General Public License. MariaDB intended to maintain high compatibility with MySQL, ensuring a drop-in replacement capability with library binary parity and exact matching with MySQL APIs and commands.

MariaDB is based on SQL and supports ACID-style data processing with guaranteed atomicity, consistency, isolation and durability for transactions. Among other features, the database also supports JSON APIs, parallel data replication and multiple storage engines.

The MariaDB software runs on Windows, Linux and macOS operating systems; it's available on Intel and IBM Power8 hardware platforms and runs as a service on several cloud platforms. Supported programming languages include C++, C#, Java, Python and others.

MariaDB sports many new speed improvements compared directly with standard MySQL. This improved performance makes MariaDB stand apart from the baseline performance of traditional MySQL servers.

**Pros**

It's available for free.

It offers a lot of functionality even for a free database engine.

There are a variety of user interfaces that can be implemented.

It can be made to work with other databases, including DB2 and Oracle.

**Cons**

You may spend a lot of time and effort to get MySQL to do things that other systems do automatically, like create incremental backups.

There is no built-in support for XML or OLAP.

Support is available for the free version, but you'll need to pay for it.

**Ideal for**: Organizations that need a robust database management tool but are on a budget.

**Microsoft SQL:** Microsoft SQL Server is a relational database management system developed by Microsoft. As a database server, it is a software product with the primary function of storing and retrieving data as requested by other software applications which may run either on the same computer or on another computer across a network (including the Internet). Microsoft markets at least a dozen different editions of Microsoft SQL Server, aimed at different audiences and for workloads ranging from small single-machine applications to large Internet-facing applications with many concurrent users.

Microsoft makes SQL Server available in multiple editions, with different feature sets and targeting different users.

SQL Server allows multiple clients to use the same database concurrently. As such, it needs to control concurrent access to shared data, to ensure data integrity when multiple clients update the same data, or clients attempt to read data that is in the process of being changed by another client. SQL Server provides two modes of concurrency control: pessimistic concurrency and optimistic concurrency. When pessimistic concurrency control is being used, SQL Server controls concurrent access by using locks. Locks can be either shared or exclusive. Exclusive lock grants the user exclusive access to the data no other user can access the data if the lock is held. Shared locks are used when some data is being read multiple users can read from data locked with a shared lock, but not acquire an exclusive lock. The latter would have to wait for all shared locks to be released.

**Pros**

It is very fast and stable.

The engine offers the ability to adjust and track performance levels, which can reduce resource use.

You can access visualizations on mobile devices.

It works very well with other Microsoft products.

**Cons**

Enterprise pricing may be beyond what many organizations can afford.

Even with performance tuning, Microsoft SQL Server can gobble resources.

Many individuals have issues using the SQL Server Integration Services to import files.

**Ideal for**: Large organizations that use several Microsoft products.

# MongoDB

## Rival 🡪 Cassandra

**Cassandra:** Apache Cassandra is a free and open source, distributed, wide column store, NoSQL database management system designed to handle large amounts of data across many commodity servers, providing high availability with no single point of failure. Cassandra offers robust support for clusters spanning multiple datacenters, with asynchronous master less replication allowing low latency operations for all clients. Cassandra offers the distribution design of Amazon Dynamo with the data model of Google's Bigtable.

**Distributed:** Every node in the cluster has the same role. There is no single point of failure. Data is distributed across the cluster (so each node contains different data), but there is no master as every node can service any request.

**Scalability:** Designed to have read and write throughput both increase linearly as new machines are added, with the aim of no downtime or interruption to applications.

**Fault-tolerant:** Data is automatically replicated to multiple nodes for fault-tolerance. Replication across multiple data centers is supported. Failed nodes can be replaced with no downtime.

**Data model:** Cassandra is wide column store, and, as such, essentially a hybrid between a key-value and a tabular database management system. Its data model is a partitioned row store with tunable consistency. Rows are organized into tables; the first component of a table's primary key is the partition key; within a partition, rows are clustered by the remaining columns of the key. Other columns may be indexed separately from the primary key.

Tables may be created, dropped, and altered at run-time without blocking updates and queries.

Cassandra cannot do joins or subqueries. Rather, Cassandra emphasizes denormalization through features like collections.

# Solr

Redis

# Bibliography

1. <https://scalegrid.io/blog/2019-database-trends-sql-vs-nosql-top-databases-single-vs-multiple-database-use/>
2. <https://db-engines.com/en/systems>
3. <https://www.youtube.com/watch?v=5Pia4UFuMKo>
4. <https://www.youtube.com/watch?v=kR8yvxZ2nqU>