sql vs no-sql

|  |  |
| --- | --- |
| sql | no-sql |
| relational database | non-relational database |
| pre-defined schema | dynamic schema |
| vertical scaling | horizontal scaling |
| ACID | CAP |
| not suited for hierarchical data store | suited for hierarchical data store |

**relational database & non-relational database**:

A relational database is one that stores data in tables. The relationship between each data point is clear and searching through those relationships is relatively easy. The relationship between tables and field types is called a schema. For relational databases, the schema must be clearly defined.

A non-relational database is any database that does not use the tabular schema of rows and columns like in relational databases. Rather, its storage model is optimized for the type of data it’s storing.

**pre-defined schema & dynamic schema**:

pre-defined schema: traditional relational databases. You must define your schema before you can add any data. Some common databases that use strict schemas are Oracle, MS SQL Server and PostgreSQL.

一張含有 文字, 螢幕擷取畫面, 螢幕 的圖片

自動產生的描述

dynamic schema: There is no need to define the schema beforehand. When data is inserted, updated, or removed, the database builds a schema dynamically. Popular dynamic schema databases include HarperDB and MongoDB.

一張含有 文字 的圖片

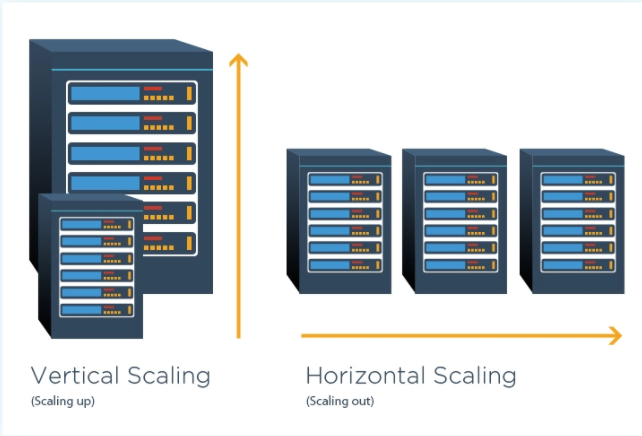
自動產生的描述

**vertical scaling &horizontal scaling**:

Scalability describes a system’s elasticity. While we often use it to refer to a system’s ability to grow, it is not exclusive to this definition. We can scale down, scale up, and scale out accordingly.

Horizontal scaling (aka scaling out) refers to adding additional nodes or machines to your infrastructure to cope with new demands. If you are hosting an application on a server and find that it no longer has the capacity or capabilities to handle traffic, adding a server may be your solution.

Vertical scaling (aka scaling up) describes adding additional resources to a system so that it meets demand.



**ACID & CAP**:

ACID: A true transaction must adhere to the ACID properties.

* Atomic: All components of a transaction are treated as a single action. All are completed or none are; if one part of a transaction fails, the database’s state is unchanged.
* Consistent: Transactions must follow the defined rules and restrictions of the database, e.g., constraints, cascades, and triggers. Thus, any data written to the database must be valid and any transaction that completes will change the state of the database. No transaction can create an invalid data state. (Note that this is different from “consistency” as it’s defined in the CAP theorem.)
* Isolated: Fundamental to achieving concurrency control, isolation ensures that the concurrent execution of transactions results in a system state that would be obtained if transactions were executed serially, i.e., one after the other. With isolation, an incomplete transaction cannot affect another incomplete transaction.
* Durable: Once a transaction is committed, it will persist and will not be undone to accommodate conflicts with other operations. Many argue that this implies the transaction is on disk as well; most formal definitions aren’t specific.

CAP: CAP is a tool to explain trade-offs in distributed systems. You can’t have all three, pick two which are CP or AP.

* Consistent: All replicas of the same data will be the same value across a distributed system.
* Available: All live nodes in a distributed system can process operations and respond to queries.
* Partition Tolerant: The system is designed to operate in the face of unplanned network connectivity loss between replicas.

**not suited for hierarchical data store & suited for hierarchical data store**:

A hierarchical database model is a data model in which the data are organized into a tree-like structure. The data are stored as records which are connected to one another through links. A record is a collection of fields, with each field containing only one value. The type of a record defines which fields the record contains.