Stay Ahead

BANNARI AMMAN INSTITUTE OF TECHNOLOGY
An Autonomous Institution Affiliated to Anna University Chennai - Approved by AICTE - Accredited by NAAC with "A+" Grade
SATHYAMANGALAM - 638 401 ERODE DISTRICT TAMIL NADU INDIA
Ph: 04295-226000 / 221289 Fax: 04295-226666 E-mail: stayahead@bitsathy.ac.in Web: www.bitsathy.ac.in

22CS405-DATABASE MANAGEMENT SYSTEMS

UNIT 5 & LP3- CAP Theorem and BASE Properties,HeidiSQL Features and Usecase

CAP Theorem and Applicability in DBMS

Let's take a detailed look at the three database management system characteristics to which the CAP theorem refers.

Consistency

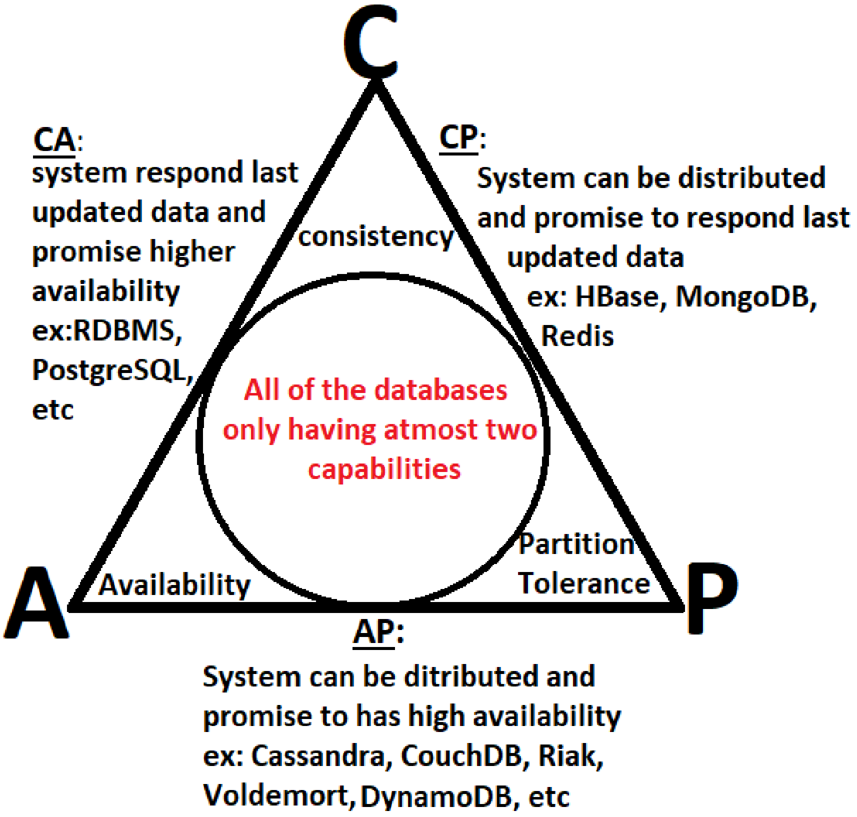
Consistency means that all clients see the same data at the same time, no matter which node they connect to. For this to happen, whenever data is written to one node, it must be instantly forwarded or replicated to all the other nodes in the system before the write is deemed 'successful.'

Availability

Availability means that any client making a request for data gets a response, even if one or more nodes are down. Another way to state this—all working nodes in the database management system return a valid response for any request, without exception.

Partition tolerance

A partition is a communications break within a database management system—a lost or temporarily delayed connection between two nodes. Partition tolerance means that the cluster must continue to work despite any number of communication breakdowns between nodes in the system.


C
CA:
system respond last updated data and promise higher availability ex:RDBMS, PostgreSQL, etc
CP:
System can be distributed and promise to respond last updated data
ex: HBase, MongoDB, Redis
consistency
All of the databases only having atmost two capabilities
Partition Tolerance
Availability
A
P
AP:
System can be ditributed and promise to has high availability
ex: Cassandra, CouchDB, Riak, Voldemort, DynamoDB, etc

- **CA(Consistency and Availability)-** The system prioritizes availability over consistency and can respond with possibly stale data.

  Example databases: Cassandra, CouchDB, Riak, Voldemort.

- **AP(Availability and Partition Tolerance)-**The system prioritizes availability over consistency and can respond with possibly stale data. The system can be distributed across multiple nodes and is designed to operate reliably even in the face of network partitions.

  Example databases: Amazon DynamoDB, Google Cloud Spanner.

- **CP(Consistency and Partition Tolerance)-** The system prioritizes consistency over availability and responds with the latest updated data.The system can be distributed across multiple nodes and is designed to operate reliably even in the face of network partitions.

Example databases: Apache HBase, MongoDB, Redis.

It’s important to note that these database systems may have different configurations and settings that can change their behavior with respect to consistency, availability, and partition tolerance. Therefore, the exact behavior of a database system may depend on its configuration and usage.

BASE PROPERTIES

The BASE properties of a database management system are a set of principles that guide the design and operation of modern databases.

The acronym BASE stands for Basically Available, Soft State, and Eventual Consistency.

Basically Available

This property refers to the fact that the database system should always be available to respond to user requests, even if it cannot guarantee immediate access to all data. The database may experience brief periods of unavailability, but it should be designed to minimize downtime and provide quick recovery from failures.

Soft State

This property refers to the fact that the state of the database can change over time, even without any explicit user intervention. This can happen due to the effects of background processes, updates to data, and other factors. The database should be designed to handle this change gracefully, and ensure that it does not lead to data corruption or loss.

Eventual Consistency

This property refers to the eventual consistency of data in the database, despite changes over time. In other words, the database should eventually converge to a consistent state, even if it takes some time for all updates to propagate and be reflected in the data. This is in contrast to the immediate consistency required by traditional ACID-compliant databases.

Uses of BASE Databases

BASE databases are used in modern, highly-available, and scalable systems that handle large amounts of data. Examples of such systems include online shopping websites, social media platforms, and cloud-based services.

Difference between Base Properties and ACID Properties

ACID	BASE
ACID (Atomicity, Consistency, Isolation, Durability) is a set of properties that guarantee the integrity and consistency of data in a traditional database.	The BASE properties are a more relaxed version of ACID that trade off some consistency guarantees for greater scalability and availability.
The primary difference between the two is that ACID requires immediate consistency,	while BASE only requires eventual consistency.
ACID is better suited to traditional transactional databases.	The BASE is more suitable for use in large-scale, highly-available systems,

HeidiSQL

HeidiSQL is free software that enables you to browse and edit data, create and edit tables, views, procedures, triggers, and scheduled events from computers running one of the database systems MariaDB, MySQL, Microsoft SQL, PostgreSQL, and SQLite.
HeidiSQL is a popular open-source database management tool for MySQL, MariaDB, Microsoft SQL Server, and PostgreSQL. It provides a graphical interface for users to interact with their databases. Here are some key properties and features of HeidiSQL:

1. Cross-Platform Compatibility:
    - HeidiSQL is a cross-platform tool, compatible with Windows, Linux, and macOS.

2. Multi-Database Support:
    - It supports various database systems, including MySQL, MariaDB, Microsoft SQL Server, and PostgreSQL.
3. Intuitive User Interface:
    - HeidiSQL offers a user-friendly and intuitive interface, making it easy for both beginners and experienced database administrators to manage and manipulate databases.
4. Query and Script Execution:
    - Users can execute SQL queries and scripts directly within the application, with syntax highlighting and code completion features.
5. Database Connection Management:
    - HeidiSQL allows users to manage multiple database connections simultaneously, providing easy navigation between different databases and servers.
6. Table and Data Management:
    - Users can create, modify, and delete database tables and manage table data using the graphical interface.
7. Import and Export Data:
    - The tool supports importing and exporting data in various formats, making it convenient for transferring data between different databases or applications.
8. SSH Tunneling:
    - HeidiSQL supports secure connections through SSH tunneling, enhancing the security of database connections.
9. Transaction Management:
    - Users can work with database transactions, ensuring data consistency and integrity.
10. Stored Procedure and Function Support:
    - HeidiSQL allows users to create, modify, and execute stored procedures and functions within the application.
11. Visual Query Builder:
    - It provides a visual query builder, helping users to construct complex SQL queries without writing the code manually.
12. Database Design Tools:
    - HeidiSQL includes tools for designing and managing database structures, such as creating and altering tables, indexes, and relationships.
13. Customization Options:
    - Users can customize the appearance and behavior of HeidiSQL according to their preferences, including themes and color schemes.
14. Open Source and Free:
    - HeidiSQL is an open-source software, and users can use it for free without any licensing fees.

Heidisql use cases:

1.Database Connection and Management:

- Connect to different database servers using various protocols (TCP/IP, SSH tunneling).
- Manage multiple database connections simultaneously.
- Browse databases, tables, and views.

2.Querying and Editing Data:

- Execute SQL queries and view results.
- Edit and modify table data directly within the application.
- Import and export data in various formats.

3.Database Structure Modification:

- Create and modify database tables, indexes, and views.
- Manage triggers, stored procedures, and functions.

4.User and Privilege Management:

- Manage user accounts and their privileges.
- Grant and revoke permissions for database objects.

5.Data Backup and Restore:

- Perform database backups and restores.
- Schedule automatic backups.

6.SSH Tunneling:

- Connect to databases securely using SSH tunneling.

7.Version Control Integration:

- HeidiSQL supports version control systems like Git and integrates with them.

8.Export and Import:

- Export database structure and data to SQL files.
- Import SQL files to create or update database structures.

9.Stored Procedure and Function Editing:

- Create, edit, and execute stored procedures and functions.

10.Customization and Themes:

- Customize the look and feel of the interface.
- Choose from different themes to personalize your experience.