: What is ACID and BASE in database theory?

by Daniel J. Power

Editor, DSSResources.COM

Data-driven and document-driven decision support systems derive functionality from data stores and databases. Beginning in the late 1990s, distributed computing facilitated deployment of new database architectures and led to post-relational databases. Storing and processing data across multiple computing nodes however creates potential problems in transaction processing. The expansion of the Internet and the need for managing larger data stores faster also led to innovation. In 2000, Eric Brewer popularized the acronym "BASE" in his keynote address at the ACM Symposium titled “Towards Robust Distributed Systems”. Brewer also explained the CAP Theorum. CAP refers to **C**onsistency, **A**vailability and **P**artition Tolerance. The theorum asserts a distributed, networked system can have only two of these three properties. How are these concepts relevant to decision support?

The traditional standard for reliable transaction processing is summarized by the acronym ACID (Atomicity, Consistency, Isolation, Durability). Decision support requires a less strict standard when static historical data is used. Real-time decision support can adhere to the eventual consistency provided by BASE. The CAP theorum explains the theoretical divide between ACID and BASE compliant databases. Understanding database theory can help decision support designers make good design choices.

Roe (2012; 2013) reviews CAP Theorum, ACID and Base. The CAP theorem states that there are three desirable system requirements for the successful design, implementation and deployment of applications in distributed computing systems. Attaining all three is not however possible. The three are:

- **Consistency** refers to predictability and reliability of data in a database across all nodes. This is the same idea of consistency described in ACID.
- **Availability** means the given system is available when needed.
- **Partition Tolerance** refers to whether a given system continues to operate even when there is partial data loss or temporary system failure or interruption. Ideally a single node failure will not cause the entire system to stop functioning.

Let's examine the ACID requirement for a database transaction system in more detail.

- **Atomicity** means either the task or tasks within a transaction are performed or none are performed (all or none rule).

- **Consistency** means the transaction meets all rules defined by the system at all times. The transaction does not violate those rules and the database must remain in a consistent state at the beginning and end of a transaction. There are no half-completed transactions.

- **Isolation:** No transaction has access to any other transaction that is in an intermediate or unfinished state. Each transaction is independent.

- Finally, **durability** means the transaction is complete and it will persist. The completed transaction will survive system failure, power loss and other types of system breakdowns.

The standards of BASE challenge some of the long held expectations for transaction processing. That's ok because decision support is not transaction processing. Let's review those standards:

- **Basically Available:** This constraint states that the system does guarantee the availability of the data as specified by the CAP Theorem -- there will be a response to any request. **But**, that response could still be a ‘failure’ to obtain the requested data or the data may be in an inconsistent or changing state.

- **Soft state:** The state of the system could change over time, so even during times without input there may be changes going on due to ‘eventual consistency,’ thus the state of the system is always ‘soft.’

- **Eventual consistency:** The system will *eventually* become consistent once it stops receiving input. The data will propagate to everywhere it should sooner or later, but the system will continue to receive input and is not checking the consistency of every transaction before it moves onto the next one.

Real-time decision support systems can perform satisfactorily is a **BASE** compliant database environment. Data warehouses have been built with denormalized, historical data for many years. In conclusion, data and document-driven DSS designers need to understand ACID and BASE and the CAP theorum, but historical data that is properly stored is ACID compliant by default. Streaming, real-time data used for decision support can meet **BASE** standards and that is a reasonable expectation. Decision support needs **AP** in **CAP** much more than consistency.

References

Brewer, E. A. (July 2000). Towards Robust Distributed Systems. ACM Symposium on the Principles of Distributed Computing. Retrieved from http://www.cs.berkeley.edu/~brewer/cs262b-2004/PODC-keynote.pdf.

Grey, J., “The Transaction Concept: Virtues and Limitations,” June 1981.

Roe, C., "ACID vs. BASE: The Shifting pH of Database Transaction Processing," Dataversity, March 1, 2012 at URL http://www.dataversity.net/acid-vs-base-the-shifting-ph-of-database-transaction-processing/

Further Reading

Bartels, D. Taking NoSQL 1.0 on a Journey into the Enterprise [1-part video]. Retrieved from http://www.dataversity.net/archives/6600.

Bloor, R. The Coming Database Revolution. Retrieved from http://www.dataversity.net/archives/5638.

DeCandia, G., Hastorun, D., Jampani, M., Kakulapati, G., Lakshman, A., Pilchin, A., Sivasubramanian, S., Vosshall, P., Vogels, W. Dynamo: Amazon’s Highly Available Key-value Store. Retrieved from http://s3.amazonaws.com/AllThingsDistributed/sosp/amazon-dynamo-sosp2007.pdf.

Ingenthron, M. How AOL Advertising Uses NoSQL to Make Millions of Smart Targeting Decisions Every Hour [1-part video]. Retrieved from http://www.dataversity.net/archives/6786.

Kerner, S.M. Inside Facebook’s Open Source Infrastructure. Retrieved from http://www.developer.com/open/article.php/3894566/Inside-Facebooks-Open-Source-Infrastructure.htm.

Hsieh, J. Cloud Deployment of Hadoop and HBase [4-part video]. Retrieved from http://www.dataversity.net/archives/6530.

Meir-Huber, M. NoSQL – The Trend for Databases in the Cloud? Retrieved from http://soa.sys-con.com/node/1615716.

Schireson, M. Re-inventing the Database: What to Keep and What to Throw Away [1-part video]. Retrieved from http://www.dataversity.net/archives/6781.

Thomas, G.J. Big Data on the Micro Scale. Retrieved from http://www.dataversity.net/archives/7287.

Thomas, G.J. Government Data in the Cloud: Things to Consider. Retrieved from http://www.dataversity.net/archives/5864.

Upton, T. A Conversation with Bruce Lindsay. Retrieved from http://queue.acm.org/detail.cfm?id=1036486.

Vogel, W. "Eventually Consistent – Revisited" covers this topic in detail.

Author: Daniel Power
Last update: 2013-10-12 06:00