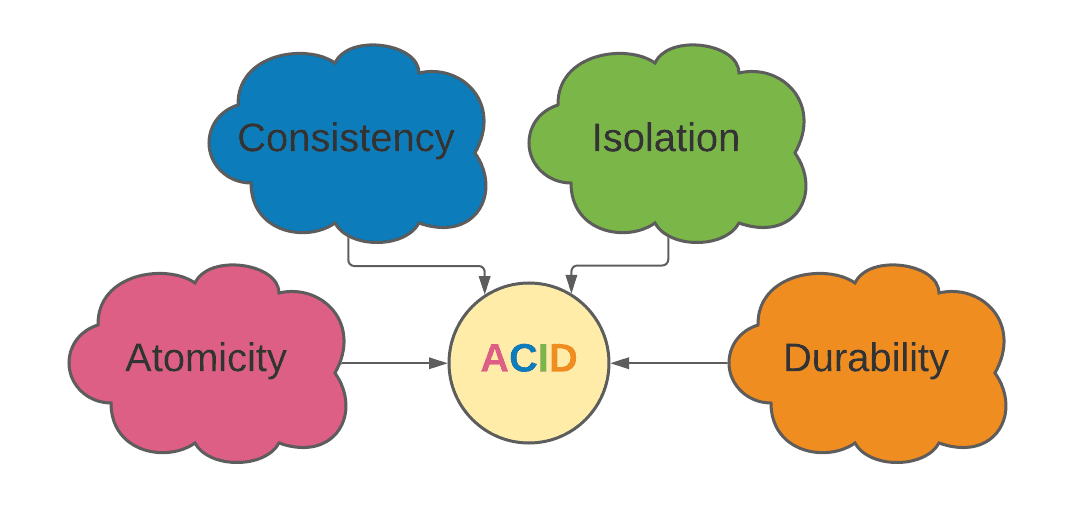
**What is ACID Properties**

In [transaction](https://www.techtarget.com/searchcio/definition/transaction) processing, ACID (atomicity, consistency, isolation, and durability) is an acronym and mnemonic device used to refer to the four essential properties a transaction should possess to ensure the [integrity](https://www.techtarget.com/searchdatacenter/definition/integrity) and reliability of the [data](https://www.techtarget.com/searchdatamanagement/definition/data) involved in the transaction. The acronym is commonly associated with relational database management systems ([RDBMSs](https://www.techtarget.com/searchdatamanagement/definition/RDBMS-relational-database-management-system)) such as [MySQL](https://www.techtarget.com/searchoracle/definition/MySQL) or [SQL Server](https://www.techtarget.com/searchdatamanagement/definition/SQL-Server), although it can apply to any system or [Application](https://www.techtarget.com/searchsoftwarequality/definition/application) that processes transactions. A transaction is a single logical unit of work that accesses and possibly modifies the content of a database. Transactions access data using read and write operations. In order to maintain consistency in a database, before and after the transaction, certain properties are followed.

**ACID Properties** Acid is an acronym that refer to the set of 4 key properties that define a transaction: 1)Atomicity 2)Consistency 3) Isolation 4)Durability



**1)Atomicity**

Atomicity ensures that a transaction is treated as a single, indivisible unit of work. Either all operations within the transaction are completed successfully, or none of them are. If any part of the transaction fails, the entire transaction is rolled back to its initial state (before the transaction started), ensuring data integrity and consistency. It involves the following two operations.

* Abort: If a transaction aborts, changes made to the database are not visible.
* Commit: If a transaction commits, changes made are visible. Atomicity is also

Know as the ‘All or nothing rule’

**Example:**

Customer X

Balance – 900

Debit to Y – 200

Balance – 700

Transaction transfer to Y

Customer Y

Balance – 600

Credit from X – 200

Balance - 800

transfer to Y Transaction failed

Customer Y

Balance – 600

Debit to X – 200

Balance - 600

Customer X

Balance – 900

Debit to Y – 200

Balance - 900

**2)Consistency**

This means that integrity constraints must be maintained so that the database is consistent before and after the transaction.it refers to the correctness of a database. referring to the example. the total amount before and after the transaction must be maintained.

**Example:**

Customer Y

Balance – 600

Credit from X – 200

Balance - 800

Transaction

Customer X

Balance – 900

Debit to Y – 200

Balance – 700

Debit to Z 300

Balance - 300

Transaction

Customer Z

Balance – 500

Credit from X – 300

Balance - 800

**3)Isolation**

Isolation ensures that multiple transactions can occur concurrently without affecting each other. Each transaction should operate independently of and be unaware of other transactions executing concurrently. Isolation levels define the extent to which the changes made by one transaction are visible to other concurrent transactions.

**Example:**

Customer Y

Balance – 600

Credit from X – 200

Balance - 800

Transaction

Customer X

Balance – 900

Debit to Y – 200

Balance – 700

Debit to Z 300

Balance - 300

Transaction

Customer Z

Balance – 500

Credit from X – 300

Balance - 800

If two operations are concurrently running on two different accounts, then the value of both accounts should not get affected. X is making T1 and T2 transaction to account Y and Z, but both are executing independently without each other. It is known as isolation.

**4)Durability**

Durability guarantees that once a transaction is committed (that is, once it is successfully completed), its changes are permanent and cannot be rolled back. This property ensures that once the transaction has completed execution, the updates and modifications to the database are stored in and written to disk and they persist even if a system failure occurs. These updates now become permanent and are stored in non-volatile memory. The effects of the transaction, thus, are never lost.

**Uses of ACID Properties**

In totality, the ACID properties of transactions provide a mechanism in DBMS to ensure the consistency and correctness of any database. It ensures consistency in a way that every transaction acts as a group of operations acting as single units, produces consistent results, operates in an isolated manner from all the other operations, and makes durably stored updates. These ensure the integrity of data in any given database.

**Advantage of ACID Properties in DBMS**

* **Data Consistency**: ACID properties ensure that the data remains consistent and accurate after any transaction execution.
* **Data Integrity**: ACID properties maintain the integrity of the data by ensuring that any changes to the database are permanent and cannot be lost.
* **Concurrency Control**: ACID properties help to manage multiple transactions occurring concurrently by preventing without interference.
* **Recovery:** ACID properties ensure that in case of any failure or crash, the system can recover the data up to the point of failure or crash.
* **Reliability**: Guarantees committed transactions survive system failures.
* **Consistency**: Enforces rules to keep data in a valid state.

**Disadvantage of ACID Properties in DBMS**

* **Performance**: The ACID properties can cause a performance overhead in the system, as they require additional processing to ensure data consistency and integrity.
* **Scalability**: The ACID properties may cause scalability issues in large distributed systems where multiple transactions occur concurrently.
* **Complexity**: Implementing the ACID properties can increase the complexity of the system and require significant expertise and resources.  
  Overall, the advantages of ACID properties in DBMS outweigh the disadvantages. They provide a reliable and consistent approach to data
* **Limited Concurrency:** ACID properties may restrict concurrency, impacting overall system throughput.
* **Recovery Complexity:** ACID properties introduce complexities in recovery and backup strategies.
* **Trade-off with Availability:** Strict adherence to ACID properties may affect system availability in certain situations.
* **Management, ensuring data integrity, accuracy, and reliability.** However, in some cases, the overhead of implementing ACID properties can cause performance and scalability issues.Therefore, it’s important to balance the benefits ACID properties against the specific needs and requirements of the system

**Conclusion**

The **ACID** properties in **DBMS** ensure reliable and robust data management:

1. **Atomicity:** Transactions are treated as indivisible units, ensuring all changes are committed or none at all.
2. **Consistency:** The database remains in a valid state before and after transactions, enforcing predefined rules and constraints.
3. **Isolation:** Concurrent transactions don't interfere with each other, preventing data inconsistencies.
4. **Durability:** Committed changes are permanently saved and survive system failures, ensuring data reliability

Adhering to these properties ensure data integrity, consistency, isolation, and durability in a database system.