Student's Name

Professor's Name

Course

Date

ACID Properties in DBMS

Organizations rely on data to track their daily activities. Data generates information that drives business decisions. Data is often robustly managed in a database that runs on a database management system (DBMS). DBMS is an intermediary between users' application programs and the database. Although the DBMS provides a platform for data sharing, security, and integration, it minimizes data inconsistencies. Inconsistencies in a database occur when versions of the same data appear in different places within a database. Hence, the need for databases is evident when the transactions in a DBMS is correct and dependable. DBMS ensures that individual databases display atomicity, consistency, isolation, and durability (ACID) to maintain reliable processing of information in databases.

Atomicity

A DBMS must ensure all parts of a transaction run as a single and indivisible work unit. This requirement maintains that database transactions should run to the end (Dutta). Thus, a database should not execute a transaction partially (Coronel). Suppose a database transaction (T) transferring $100 from account X to Y requires two SQL statements, T^1 and T^2, to execute. Therefore, the atomicity in this system can be expressed as follows:

Balance in X before T = $600

Balance in Y before T =$300

$$T^1\ read(X)$$

$$X: X - 100$$

$$Write\ (X) = 500$$

Thus, after X completes, the balance will be:X: 500

$$T^2\ Read(Y)$$

$$Y: Y + 100$$

$$Write(Y) = 400$$

Suppose the transaction fails after T^1 executes but before completion of T^2. In that case, the balance of X will be deducted but not added to Y. This transaction produces an inconsistent database state. Hence, the DBMS will roll back the transaction to maintain atomicity.

Consistency

A database should maintain its consistency state. Therefore, a database integrity constraint and business rules should not be violated after a transaction (Coronel). When a transaction completes, a database should be in a consistent state.

Referring to the previous calculations, the total balance before and after the transaction should be consistent, as shown in the calculations:

Balance before T executes $\underline{\boldsymbol{= 600 + 300 = 900}}$

Balance after T executes $\underline{\boldsymbol{= 500 + 400 = 900}}$

Evidently, the database is consistent. Inconsistency happens when T1 runs successfully but T2 fails. For that matter, T is incomplete.

Isolation

Data used in one transaction (T^1) cannot be used in a second transaction (T^2) until (T^1) finishes. In essence, if (T^1) is accessing data item X, then X cannot be released to other transactions until

(T^1) ends (Coronel). Isolation property of databases' transactions is critical in a multiuser database environment where several database consumers update one database concurrently.

Consider the following transactions in a database:

$Let\ A = 800, B = 800$

Two transactions T^1 and T^2 can run as shown in Table 1.

Table 1

Sample Database Snapshot

$\mathbf{T^1}$	$\mathbf{T^2}$
Read (A)	Read (A)
A: =A*100	Read(B)
Write (A)	C: =A+B
Read(B)	Write(C)
B: B-100	
Write (B)	

Suppose T^1 executed until Read(B) and then T^2 starts, then interleaving of the transactions. T^2 reads correct values of A but incorrect value of B. Thus, the incorrect values of B and the sum will be computed in the following calculation:

$$T^2: (C: A(80000) + B(800) = 80800)$$

The value is inconsistent with the sum at the end of transaction:

$$T: (A + B: A(80000) + B(700) = 80700)$$

The results in the database is erroneous because of a loss of 100 units. Overall, isolation requires that transactions must run in isolation and changes should be visible after being saved into the main memory.

Durability

Databases' transactions should be reliable. Hence, the durability property ensures that updates and modifications must be stored permanently after a transaction has run to completion and must be stored permanently even if a system fails (Coronel; Dutta). For that matter, the changes become permanent because they are located in the main memory. DBMSs store transactions in a log to preserve durability when a system fails or crashes.

In brief, transactions in databases must maintain the ACID properties. DBMS are vital in preserving the properties to ensure that data and information are reliable and dependable. Database engineers should always strive to check whether the DBMS is configured to avoid data loss or errors.

Works Cited

Coronel, Carlos M. *Database systems: Design, Implementation, & Management*. Cengage Learning, 2018.

Dutta, Bikash. "ACID Properties in Database Transactions." *WTMatter*, 13 July 2020. wtmatter.com/acid-properties-database-transactions. Accessed 29 Oct. 2020.