David Thomsen

CSI 300: Problem Set 5

Source: <https://www.freecodecamp.org/news/acid-databases-explained/>

1. Lookup the definition of atomic (with regards to database transactions). Copy and paste it. Cite your source. Then rewrite the definition in your own words and give a real-world example of something that must be atomic (i.e. something that is abstract but not a database - ex. death is atomic - you’re either fully dead or you’re not dead at all).

Direct Quote: “Atomicity simply means that all queries in a transaction must succeed for the transaction to succeed. If one query fails, the entire transaction fails.”

Rewritten: Atomicity is the process of designing a database in such a way that every transaction must succeed in order for any part of it to run. If even one part of the transaction fails, the whole query will fail.

Example: A process that would be defined as atomic would be a bank transaction. If you are trying to withdraw money from your account, but fail any of the previous checks, the bank will kill the transaction entirely. You either get your money or you don’t.

-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-

Source: <https://database.guide/what-is-acid-in-databases/>

2. Lookup the definition of consistent (with regards to database transactions). Copy and paste it. Cite your source. Then rewrite the definition in your own words and give a real-world example of something that must be consistent.

Direct Quote: This ensures that you guarantee that all data will be consistent. All data will be valid according to all defined rules, including any constraints, cascades, and triggers that have been applied on the database.

Rewritten: Consistency ensures that identical queries return identical responses. This ensures that there are no outliers or funky entires in the database that may cause issues when updating or editing.

Example: One example of this is a library database. The customer will have a library card used to log books they take out. The inventory has to reflect these books being removed from the database to ensure another customer doesn’t come looking for it. If there is inconsistency within this information, both customers and staff will be confused.

-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-

Source: <https://database.guide/what-is-acid-in-databases/>

3. Lookup the definition of isolated (with regards to database transactions). Copy and paste it. Cite your source. Then rewrite the definition in your own words and give a real-world example of something that must be isolated.

Direct Quote: Guarantees that all transactions will occur in isolation. No transaction will be affected by any other transaction. So a transaction cannot read data from any other transaction that has not yet completed.

Rewritten: This is a very complicated way of saying that if two people query the same database at the same time, their transactions won't interact at all and they should have two independent versions of the DB to view. Isolation ensures that multiple transactions occurring at the same time should not impact

Example: If there is a One-of-a-Kind Joe Eastman action figure, and two people place an online order at the exact same time, both will be shown the isolated version of the site with the item available.

Whoever places the order first will receive an update saying their order has been processed, and the other person will see the site updated to reflect this item being out of stock. Should the second person continue with the order before it updates, the order will just be aborted and the second user will be informed of this.

-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-

Source: <https://www.freecodecamp.org/news/acid-databases-explained/>

4. Lookup the definition of durable (with regards to database transactions). Copy and paste it. Cite your source. Then rewrite the definition in your own words and give a real-world example of something that must be durable.

Direct Quote: Durability is a guarantee that changes made by a committed transaction must not be lost. All committed transactions must be persisted on durable, non-volatile storage, that is on disk. This ensures that any committed transactions are protected even if the database crashes.

Rewritten: Durability is a way to guarantee that commits made to the database are kept secure even if the database crashes. Durability is not a cure-all however, and the data is still able to be destroyed locally.

Example: A live example of this would be making a commit to github. If a commit is made to github, but it is not pushed, the data will remain in the commit history, even if there are additional updates added later. If the device is turned off

-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-