

 100 XP

Choose the right consistency level

3 minutes


Each of the consistency models can be used for specific real-world scenarios. Each provides precise availability and performance tradeoffs and is backed by comprehensive SLAs. The following simple considerations help you make the right choice in many common scenarios.

Configure the default consistency level

You can configure the default consistency level on your Azure Cosmos DB account at any time. The default consistency level configured on your account applies to all Azure Cosmos DB databases and containers under that account. All reads and queries issued against a container or a database use the specified consistency level by default.

Read consistency applies to a single read operation scoped within a logical partition. The read operation can be issued by a remote client or a stored procedure.

Guarantees associated with consistency levels

Azure Cosmos DB guarantees that 100 percent of read requests meet the consistency guarantee for the consistency level chosen. The precise definitions of the five consistency levels in Azure Cosmos DB using the TLA+ specification language are provided in the [azure-cosmos-tla](#)  GitHub repo.

Strong consistency

Strong consistency offers a linearizability guarantee. Linearizability refers to serving requests concurrently. The reads are guaranteed to return the most recent committed version of an item. A client never sees an uncommitted or partial write. Users are always guaranteed to read the latest committed write.

Bounded staleness consistency

In bounded staleness consistency, the reads are guaranteed to honor the consistent-prefix guarantee. The reads might lag behind writes by at most "K" versions (that is, "updates") of an

item or by "T" time interval, whichever is reached first. In other words, when you choose bounded staleness, the "staleness" can be configured in two ways:

- The number of versions (K) of the item
- The time interval (T) reads might lag behind the writes

For a single region account, the minimum value of K and T is 10 write operations or 5 seconds. For multi-region accounts the minimum value of K and T is 100,000 write operations or 300 seconds.

Session consistency

In session consistency, within a single client session reads are guaranteed to honor the consistent-prefix, monotonic reads, monotonic writes, read-your-writes, and write-follows-reads guarantees. This assumes a single "writer" session or sharing the session token for multiple writers.

Consistent prefix consistency

In consistent prefix, updates made as single document writes see eventual consistency. Updates made as a batch within a transaction, are returned consistent to the transaction in which they were committed. Write operations within a transaction of multiple documents are always visible together.

Assume two write operations are performed on documents *Doc 1* and *Doc 2*, within transactions T1 and T2. When client does a read in any replica, the user sees either "*Doc 1* v1 and *Doc 2* v1" or "*Doc 1* v2 and *Doc 2* v2", but never "*Doc 1* v1 and *Doc 2* v2" or "*Doc 1* v2 and *Doc 2* v1" for the same read or query operation.

Eventual consistency

In eventual consistency, there's no ordering guarantee for reads. In the absence of any further writes, the replicas eventually converge.

Eventual consistency is the weakest form of consistency because a client may read the values that are older than the ones it read before. Eventual consistency is ideal where the application doesn't require any ordering guarantees. Examples include count of Retweets, Likes, or nonthreaded comments

Next unit: Explore supported APIs

[Continue >](#)
