

# Consistency, Fault Tolerance, and Availability

This page describes how Xpand's architecture is designed for Consistency, Fault Tolerance, and Availability.

- [Consistency](#)
- [Fault Tolerance](#)
- [Availability](#)
  - [Group Membership and Quorum](#)
  - [Partial Availability](#)
  - [Availability Requirements](#)

See also the System Administration Guide on [Fault Tolerance and High Availability](#)

## Consistency

Many distributed databases have embraced *eventual* consistency over *strong* consistency to achieve scalability. However, eventual consistency comes with a cost of increased complexity for the application developer, who must develop for anomalies that may arise with inconsistent data.

Xpand provides a consistency model that can scale using a combination of intelligent [data distribution](#), [multi-version concurrency control \(MVCC\)](#), and Paxos. Our approach enables Xpand to scale writes, scale reads in the presence of write workloads, and provide strong ACID semantics.

For an in-depth explanation of how Xpand scales reads and writes, see [Concurrency Control](#).

Xpand takes the following approach to consistency:

- Synchronous replication within the cluster. All nodes participating in a write must provide an acknowledgment before a write is complete. Writes are performed in parallel.
- The [Paxos](#) protocol is used for distributed transaction resolution.
- Xpand supports for Read Committed and Repeatable Read (Snapshot) isolation levels with limited support for Serializable.
- Multi-Version Concurrency Control (MVCC allows) for lockless reads and ensures that writes will not block reads.

## Fault Tolerance

Xpand provides fault tolerance by maintaining multiple copies of data across the cluster. By default, Xpand can accommodate a single node failure and automatically recover with no loss of data. The degree of fault tolerance (nResiliency) is configurable and Xpand can be set up to handle multiple node failures and zone failure.

For more information, including how to adjust fault tolerance in Xpand, see [Understanding Fault Tolerance](#), [MAX\\_FAILURES](#), and [Zones](#).

## Availability

In order to understand Xpand's availability modes and failure cases, it is necessary to understand our group membership protocol.

### Group Membership and Quorum

Xpand uses a distributed group membership protocol. The protocol maintains two fundamental sets:

1. The static set of all nodes known to the cluster
2. The set of nodes that can currently communicate with each other.

The cluster cannot form unless more than half the nodes in the static membership are able to communicate with each other (a [quorum](#)).

For example, if a six-node cluster experiences a network partition resulting in two sets of three nodes, Xpand will be unable to form a cluster.



However, if more than half the nodes are able to communicate, Xpand will form a cluster.



For performance reasons, MAX\_FAILURES defaults to 1 to provide for the loss of one node or one zone.

## Partial Availability

In the above example, Xpand formed a cluster because a quorum of nodes remained. However, such a cluster could offer only partial availability because the cluster may not have access to the complete dataset.

In the following example, Xpand was configured to maintain two replicas. However, both nodes holding replicas for A are unable to participate in the cluster (due to some failure). When a transaction attempts to access data on slice A, the database will generate an error that will surface to the application.



## Availability Requirements

Xpand can provide availability even in the face of failure. In order to provide full availability, Xpand requires that:

- A majority of nodes are able to form a cluster (i.e. quorum requirement).
- The available nodes hold at least one replica for each set of data.