# Change Feed Processor tool

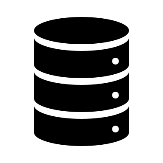
## Abstract

The tool represents a host for distributing change feed events in partitioned collection across multiple observers. Instances of the host can scale up (by adding) or down (by removing) dynamically, and the load will be automatically distributed among active instances in about-equal way.

## When to use

The tool is targeted for scenarios when single host is not fast or powerful enough to process changes in monitored collection that come at high rate. It can also be used to monitor changes when rate is not high in partitioned collection, although that can be done without using auxiliary collection for leases and without observers.

## Design highlights



Monitored collection

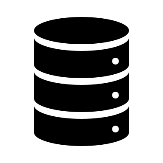
…..I” 0.05"

“city”: “O..S”..I” 0.05"

“city”: “J..N”..I” 0.05"

“city”: “F..I”..I” 0.05"

“city”: “A..E” 0.05"



{ “Owner”: “Host1” “Continuation”: “4” }

{ “Owner”: “Host1” “Continuation”: “5” }

{ “Owner”: “Host2” “Continuation”: “3” }

{ “Owner”: “Host2” “Continuation”: “2” }

Host1

Consumer 1

Consumer 2

Host2

Consumer 1

Consumer 2

Lease collection

* The changes are happening in the “Monitored collection”, which is partitioned by “city”. The arrows indicate current position (continuation) in the change feed.
* For housekeeping, another collection, “Lease collection” is used.
  + There is one lease per partition.
  + Lease has the following attributes:
    - Owner: specifies the host that owns the lease (no owner – the lease is available).
    - Continuation: specifies position (continuation token) in the change feed for particular partition.
    - Timestamp: last time lease was updated.
    - Expiration interval: if lease wasn’t updated during this time it is considered expired and can be taken by other hosts.
* Each instance of Change Feed Host does the following:
  + Determine how many leases are its target. This is based on how many other instances of hosts own active leases.
  + Acquire leases: all available leases, then expired leases, then if still the number of own leases is less than the target, it starts to steal leases from other hosts.
  + Renew leases: the host periodically renews its leases. This makes sure that the leases are active / not expired.
  + For each lease, there is an instance of worker that does the following:
    - Read change feed for partition the lease is associated with.
    - Notifies consumer (observer, one per lease, or worker) about these changes.
    - If lease is lost or stolen, it notifies the observer.
    - For each read, checkpoints last continuation token to its lease.
  + For concurrency safety, for each lease update, ETag (if-match condition) is used. If it fails, the host reads the lease document and checks whether it still owns it, and if not, the lease is considered lost.
  + On shutdown, releases all leases (set Owner to null), but keeps the Continuation so that next time changes when change feed for this partition is read, it will continue from last checkpoint.
* Scalability considerations
  + If one host owns all leases, and another one comes up, it will start stealing leases from first one, so that eventually the leases will be about-equally divided between running hosts.
  + If a host dies, its leases will expire soon and will be distributes between other hosts.

## See also

* Blog: [Introducing Change Feed support in Azure DocumentDB](https://azure.microsoft.com/en-us/blog/introducing-change-feed-support-in-azure-documentdb/).
* Article: [Working with the Change Feed support in Azure DocumentDB](https://azure.microsoft.com/en-us/blog/introducing-change-feed-support-in-azure-documentdb/).