High availability and recovery in IBM Db2 Warehouse on Cloud

Aniket Kulkarni, IBM
Littleton, Massachusetts USA
email: aniket.kulkarni@us.ibm.com

Venkatesh Gopal, IBM
Leawood, Kansas USA
email: gopalv@us.ibmcom

Abstract

One of the key metrics that cloud services are measured by is availability. Cloud service providers need to architect their services to mitigate the risk of underlying component failures, which increases as the sizes and complexities of the services grow.

High availability describes a service's abilities to handle these risks. At the most basic level, high availability (HA) is achieved with the ability to (a) detect component failures, and (b) recover quickly from them in an automated fashion. While container orchestration and management systems, such as Kubernetes, enable these abilities to an extent, there are various ways to build upon them to ensure even higher levels of reliability in specific types of services.

In this paper, we focus on how high availability and reliability is achieved in IBM® Db2® Warehouse on Cloud, the cloud data warehouse from IBM. We specifically highlight its containerized service model, multi-tier reliability, and ability to recover from abject failures.

Multi-tier reliability

Db2 Warehouse on Cloud offers clients a shared-nothing, massively parallel cloud data warehouse, deployed as a containerized service[1] model built on the IBM Cloud™ Container service. The IBM Cloud Container service is itself built on top of Kubernetes[2], an open-source container-orchestration system for automating deployment, scaling and management of containerized applications.

To support high availability and reliability in Db2 Warehouse on Cloud, IBM built upon the Kubernetes foundation with a multi-tier reliability design that includes:

- Tier 1: No single point of failure in the system, every component has one or more redundant copies and automatic recovery is initiated if multiple redundant components fail
- Tier 2: Automatic intra-container recovery if components within a container fail
- Tier 3: Automatic container recovery if a container itself fails
- Tier 4: Automatic recovery of the service if the server(s) hosting the container(s) encounter a system failure

In addition, Db2 Warehouse on Cloud provides user-defined and controlled rapid backup and restore functionality. The service also has 24x7 DevOps and monitoring support that tracks failures and system health in the event of multiple or unrecoverable failures that even the multi-tier reliability layer cannot handle.

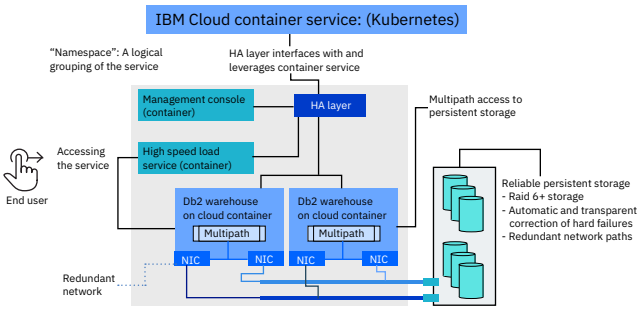

Figure 1: A high-level view of the architecture behind the reliability of the Db2 Warehouse on Cloud service

- Each component in the service is a container
- The HA layer extends and augments Kubernetes to provide faster HA
- Network connections to persistent storage are redundant and logically grouped by a multipath
- The HA layer can recover containers if components inside the container fail, without restarting the container, making recovery quick
- The persistent storage has multiple redundancies at network and storage level

Let's take a look at each of these tiers as well as the backup and restore capabilities in greater depth.

Reliability Tier 1: Component redundancy

Db2 Warehouse on Cloud is an elastic data warehouse, and allows users to independently scale CPU cores ("compute") and storage capacity on demand to meet changing business requirements. The architecture that enables this elasticity is built by decoupling compute and storage. Compute capacity is managed by the IBM Cloud Container Service built on top of Kubernetes and the storage is placed in network-attached, high-performance IBM Cloud block storage.3

Because the network itself is such an important component for reliability and performance, IBM also ensured that there is no single point of failure for the network. Each of the underlying servers has a pair of redundant network interface controllers (NICs) connected to separate paths which lead to the storage backplane and are bonded to ensure high performance (through load balancing) when possible. In addition, the storage backplane has a pair of redundant "targets," or access points, that allow network access to the storage infrastructure. On the compute (container) side, an active/passive multipath configuration ensures interruption-free "failover," such that if one of the storage paths fails, the system switches to the other one immediately.

In addition to the above, the storage backend utilizes multiple redundant and hot swappable SSDs in a RAID 6+ configuration. This is an improvement over traditional RAID 6, and allows management and repair of any problems in the SSDs without interruption.

Reliability Tier 2: Intra container recovery

In a traditional Kubernetes or web services model, the failure of any component within a container is handled by restarting the entire container. While simple and effective, this is insufficient for supporting users that demand a high level of uptime for their business-critical applications.

Considering the nature of the workloads in a cloud data warehouse, where large datasets are involved, it is faster to recover a single failing component of the container rather than to restart the entire container.

A full restart would require the container to dissociate storage, give up compute, select new compute, and re-associate storage. In contrast, directly addressing the affected component provides key time savings that keeps Recovery Time Object (RTO) to a minimum.

This shortcoming in the traditional model is addressed by splitting such recoveries into two tiers—Tier-2 and Tier-3. In Db2 Warehouse on Cloud, the "HA Layer" extends into each of the containers, aptly named the "Container HA Layer." The Container HA Layer tracks the health of each of the components within the container, and takes corrective recovery actions if any of these components fails.

For example, suppose a Db2 container encounters a failure within one of its data partition processes. The Container HA Layer for that container will detect this failure and attempt to restart those data partitions, followed by database recovery to ensure data consistency. Furthermore, if there is a cascade of failures, such as the failure of one data partition while another is in the process of being recovered, then the Container HA Layer will abort the recovery in progress and do a clean recovery of both partitions together.

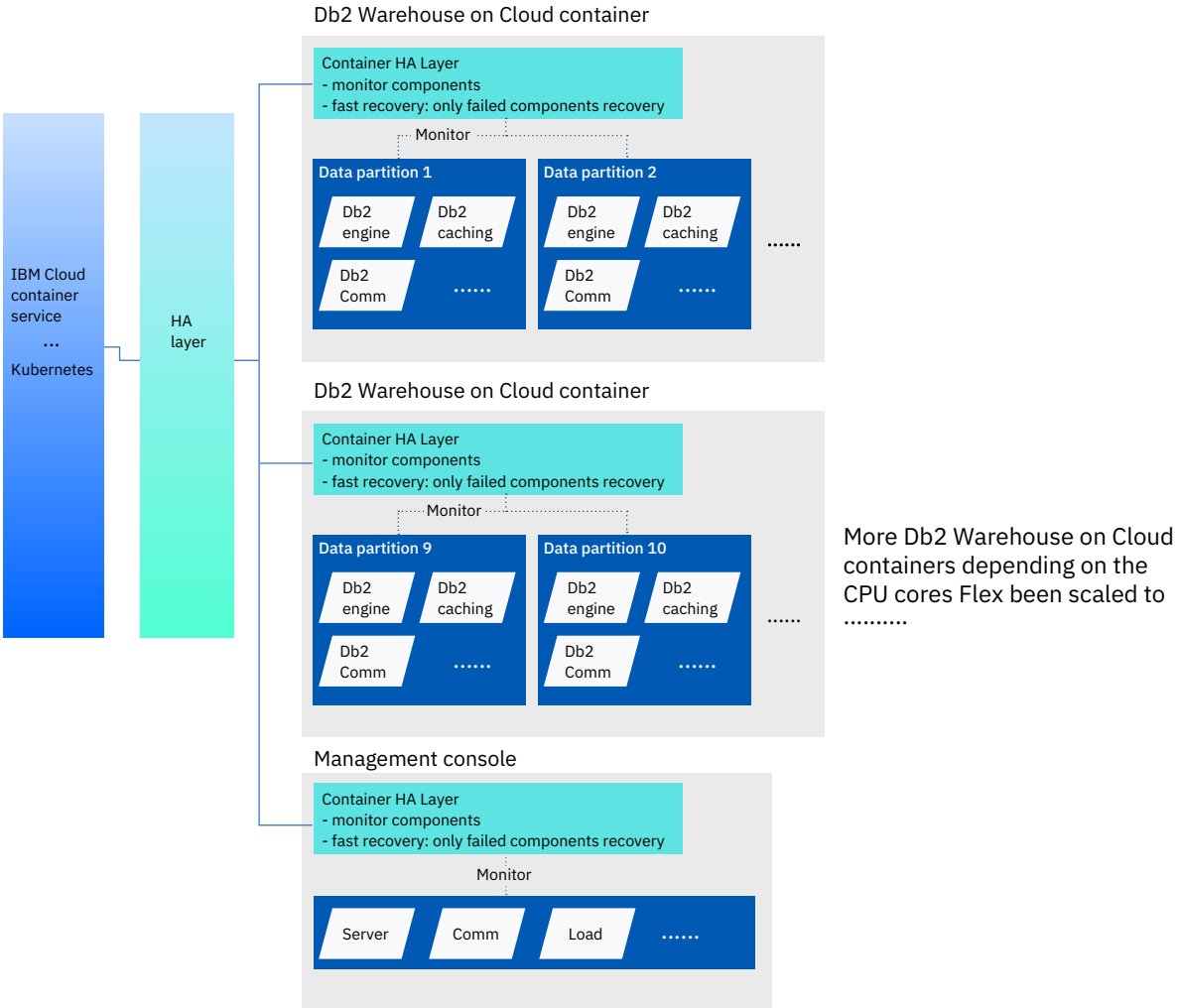

Figure 2: Tier-2 intra-container HA:

- Each container has an HA Layer embedded in it that tries to provide a fast HA if only a permissible subset of components fails within that container
- For Db2 Warehouse on Cloud containers the HA Layer monitors each of the partitions and components within each partition. If a subset of partitions fails, only those partitions are automatically restarted and recovered
- For Non-Db2 Warehouse on Cloud containers, each component within the container is similarly monitored and recovered if possible
- The HA Layer interfaces with and extends the Container Service's HA Layer. That lets it escalate to Tier-3 recovery if Tier-2 recovery is not possible

Reliability Tier 3: Container recovery

If any of the containers encounters multiple errors, or the container itself is terminated, then Tier-3 recovery will initiate. In this case, the HA Layer falls back onto the traditional Kubernetes HA model of container restarts, and:

1. The affected container(s) are cleanly stopped and cleaned up.
2. The associated persistent storage is dissociated, and the container service then reschedules the container onto a new set of compute cores.
3. At that time, the storage is automatically re-associated, and the container starts up.

Note that for a multi-container service like Db2 Warehouse on Cloud that is comprised of more than one container, the HA Layer coordinates a simultaneous database recovery across all containers.

Aside from being the fallback path for Tier-2 reliability, this recovery also handles another important use case for data integrity in the event of multiple failures at the network level. Specifically, in the rare case that there are multiple network failures on each of the redundant network paths, then to protect data consistency, Db2 Warehouse on Cloud will automatically put the storage in a protected read-only mode.

If the HA Layer detects this situation, it will force a container stop, dissociate, reschedule, associate and restart the container. This will move the container to another underlying server with a healthy network and get the storage out of the protected mode, thereby preserving data integrity and successfully recovering the system.

Reliability Tier 4: Service recovery

Container recovery is the most traditional Kubernetes HA model. The HA Layer enables Kubernetes to detect total failures of the underlying servers that provide the compute cores to the containers.

The recovery action is quite like Tier-3: all containers are stopped, all affected storage is disassociated, all affected containers are rescheduled, all storage is associated, and all affected containers are started again.

Any server that fails or becomes unresponsive is detected by the HA layer by a series of missed heartbeats. After a grace period, recovery actions (mentioned above) are taken, and the server is marked as “un-schedulable” and won’t be used to start any other containers. This gives operations teams the opportunity to service the affected server and then reinstate it without causing additional disruption to the end user.

Backup and restore

The ability for users to perform database backups and restores is crucial for protecting the cloud service against user errors, and in extremely rare cases, as a last resort to recover from a total failure. Db2 Warehouse on Cloud gives users a simple interface to manage this functionality themselves:

- The user can schedule backups to run when it’s the most convenient for their business. The backups will run once every 24 hours on the set schedule.
- The last seven backups are retained and, if needed, users can restore the database from one of these backups quickly with a single button click.

Redirect-on-Write Snapshots: The key to lightning fast backup and restore

The reliable, persistent storage layer also supports high speed, near instant, redirect-on-write snapshots for backups. Each data partition in the database is backed by a network-attached, reliable, persistent storage volume. Figure 3 shows the time taken (in minutes) for a snapshot-based backup is dramatically less than traditional database backup technologies for a 3 TB – 4.2 TB database. The snapshot backups finish in mere minutes (between 1-3 minutes), whereas traditional database backups take anywhere from 2 to 3.5 hours.

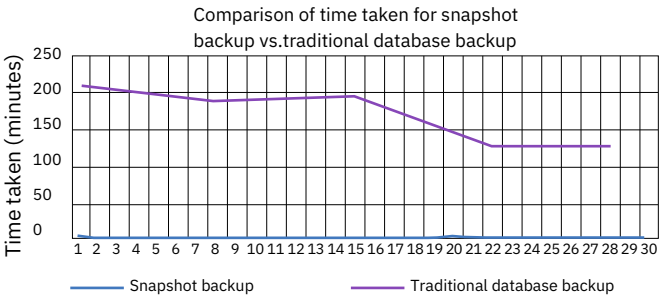

Figure 3: This plot compares time taken for backups (minutes) for a database of size between 3 TB – 4.2 TB over the course of a month, using snapshot backups and traditional database backups. This data was measured across multiple systems running production workloads.

ROW is an optimal variant of Copy-on-Write (COW) snapshot technology and is used for reliable, persistent storage. The data blocks are strung together by references, and the snapshot is merely a group of references that allow the reader to read the right set of blocks.

For example, if the user takes a snapshot and then changes a block in the volume, the storage volume will write the changes to a new location and update the references to point to the new location as the current version of that block. The snapshot will continue to refer to the earlier block, while the rest of the blocks remain unchanged and are referred to by both the snapshot and the current view.

Db2 Warehouse on Cloud "pauses" the system workload for a minute or so and uses that quiesced status to take a snapshot of the volume before "un-pausing" the workload. The most recent 7 backups are retained.

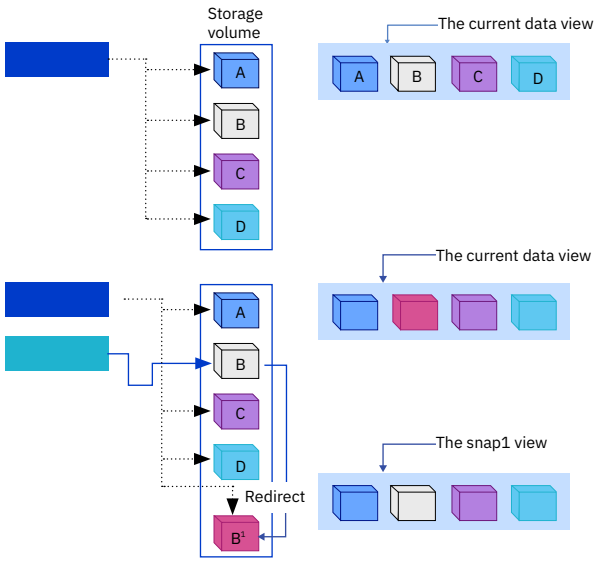

Figure 4: This figure illustrates the ROW operation. The volume uses references to tie the most current version of blocks and let the end user read that.If a snapshot (Snap-1) is created and then B is changed to B' then the original blocks remain unchanged and B' is written at a new location.

Snap-1 now points to original B block while the current view of data starts point to B'If a snapshot (Snap-1) is created and then B is changed to B' then the original blocks remain unchanged and B' is written at a new location.

Snap-1 now points to original B block while the current view of data starts point to B'

This strategy, along with ROW gives the user the flexibility of "time travel restore." That is, the user can restore to Snapshot 1, while preserving Snapshots 2 through 7, enabling the user to perform a separate restore to one of those snapshots if needed.

Learn More

A guided product tour is also available if you'd like to interact with the Db2 Warehouse on Cloud's interface firsthand.

IBM Corporation
New Orchard Road
Armonk, NY 10504

Produced in the United States of America
December 2018

1 "IBM Cloud Kubernetes Service."
https://www.ibm.com/cloud/container-service

2 "Production-Grade Container Orchestration."
https://kubernetes.io/

3 "Block Storage."
https://www.ibm.com/cloud/block-storage

Please Recycle

XXX00000-USEN-00