

Custom Object Storage with Consistent Search Store

Abstract:

Organizations with multiple teams generally end up with different applications that manage its own database interactions. This approach would be fine at a smaller scale but as we move forward towards a large scale, the situation will become more error-prone. Because now each application needs to either deploy experts in each team or build expertise to make the data available, consistent, and usable at all times. This leads to a waste of resources both in terms of computing as well as human. In addition to that, applications need to manage Disaster Recovery (DR), migrations (if needed), Recovery Time Objective (RTO), and Recovery Point Objective which all lead to inconsistencies. Hence, we developed a strongly consistent centralized document service with a consistent secondary search store.

Outline:

1. What's the need?

Many modern organizations are trying to shape the global market. And to be at the top of the game, it is required to build and deliver consistent features/products. Hence, many teams are managing different services and as per their use cases, they have different database needs. And because of that what modern organizations generally end up with is many scattered data services, and each team/project owner has to maintain their own resources. The proposed solution is meant to solve these problems:

- Database abstraction
- Connection management
- Achieve multi-tenancy
- Maintaining RTO and RPO as per agreement and maintenance
- Centralized alerting and monitoring
- Maintain required SLA (With the support of distributed caching and other features)
- Consistent Search Store

2. Requirements:

Through careful evaluation, the following requirements are specified for the proposed solution:-

- Data Model: supports hierarchical data structures and collections in a hierarchy. Store your data in documents, organized into collections.

- Query: queries can be used to retrieve individual, specific documents or to retrieve all the documents in a collection that match your query parameters. Queries can include multiple, chained filters and combine filtering and sorting.
- strong consistency guarantees, atomic batch operations, and real transaction support (transaction support will be optional and can be configured based on project requirements).
- Transactionally consistent local secondary indexes.
- Multitenancy
- Real-Time change data capture for near-real-time processing.
- Transactional writes for closely interrelated (co-partitioned) data.
- Consistent Schema checks.
- Partial update of an object
- Consistent Search store implementation
- Auditing all updates to ease debugging

3. What we achieved with the proposed solution.

- Data governance :
 - Centralized service to maintain different requirements of different projects. Hence ease of development for different teams.
 - Consistent document store
 - Highly available
 - Automated system to cater to RTO/RPO
 - Eventual consistent search store
- Database abstraction :
 - Can communicate with an application of any language which supports Protobuf communication
 - Make database communication simple
- Maintenance :
 - Automated APIs to increase/reduce infrastructure resource
 - Consolidated Monitoring, Alerting and Auditing to observe the overall infrastructure