Hybrid Index v2

Chetan Mehrotra | @chetanmeh | Oakathon - August 2017

Bē
David Mascha

Hybrid Index v1

- OAK-4412 - Oak 1.6
- Near Real Time Indexing Support
- Query Performed as union of
  - Per Cluster Node Transient Lucene Index
  - Persisted Lucene Index
- Transient index pruned with every Async indexer run
- Search latency reduced ~ 1s

Property Indexes

- Benefits
  - Synchronous Index
  - Uniqueness Constraints
- Drawbacks
  - Poor performance over remote storage
  - Prone to conflicts
  - Storage Overhead

Hybrid Index v2

- OAK-6535
- Proposal - Synchronous Lucene Property Indexes
- Supports
  - Sync Indexing
  - Unique Indexes
- Uses Property index as a "transient" sub index
- Property indexes periodically pruned
- Query Performed as union of
  - "transient" property index
  - Persisted Lucene Index

Index Definition

```
/oak:index/assetType
  - jcr:primaryType = "oak:QueryIndexDefinition"
  - type = "lucene"
  - async = ["async", "nrt"]
  + indexRules
    + nt:base
      + properties
        + resourceType
          - propertyIndex = true
          - name = "assetType"
          - sync = true
```

Sync Index

```
/oak:index/uuid
  - jcr:primaryType = "oak:QueryIndexDefinition"
  - type = "lucene"
  - async = ["async", "nrt"]
  + indexRules
    + nt:base
      + properties
        + uuid
          - propertyIndex = true
          - name = "jcr:uuid"
          - unique = true
```

Unique Index

Sync Indexes - Storage

- One *sub* property index per **sync** property definition
- Indexed values stored in buckets
- Buckets switched on every Async Indexer Run
- Only 2 buckets kept
- Older buckets removed via periodic job

```
/oak:index/assetType
  + :data   //Stores the lucene index files
    + segments.gen
    + _13x.cx
      - jcr:data = //Lucene Index Files
  + :property-index
    + resourceType
      - head = 2                  //Current active bucket
      - previous = 1
      + 1
        - jcr:created = 1502274302 //creation time in millis
        - lastUpdated = 1502284302
        + type1                   //Indexed value
          + libs                  //content mirror storage
            + login
              + core
                - match = true
        + <value>
          + <mirror of indexed path>
      + 2
        - jcr:created = 1502454302
        + type1
          + ...
      + 3
        - jcr:created = 1502154302
        + type1
```

Sync Indexes - Read/Write/Delete Flow

- Write Flow
  - ContentMirrorStoreStrategy layout
  - Current head bucket used for Index Storage Node
- Read Flow
  - For queries involving property constraint on sync properties
  - Union Cursor Created on
    - Cursor from head and previous bucket
    - Cursor from Lucene Index
  - In case multiple sync properties in same query - Select one based on cost
- Pruning
  - Change head and previous bucket post each async run
  - Remove any other bucket

Unique Indexes - Storage

- One *sub* property index per unique property definition
- "older" entries periodically removed

```
/oak:index/assetType
  + :data   //Stores the lucene index files
  + :property-index
    + uuid
      + <value 1>
        - entry = [/indexed-content-path]
        - jcr:created = 1502274302 //creation time in millis
      + 49652b7e-becd-4534-b104-f867d14c7b6c
        - entry = [/jcr:system/jcr:versionStorage/63/36/f8/...]
        - jcr:created = 1502274302
      + ffaabe-becd-4534-b104-f867d14c7b6c
        - entry = [/jcr:system/jcr:versionStorage/aa/12/ca/...]
        - jcr:created = 1502214302  //Old value. To be removed
```

Unique Indexes - Read/Write Flow

- Write Flow checks
  - Entry in unique property index
  - Entry in Lucene index via Lucene query
- Read Flow
  - For queries involving property constraint on unique properties
  - Union Cursor Created on
    - Cursor from Lucene Index
    - Cursor from head and previous bucket
  - In case multiple sync properties in same query - Select one based on cost
- Pruning
  - Remove entries older than last async indexer run via traversal

Points to note

- Queries involving sorting would not use sub property indexes
- Hybrid Index v2 may replace
  - all property indexes - Like /oak:index/slingResourceType
  - unique index - LIke /oak:index/uuid
  - most of nodetype index - Except nodetype index on oak:QueryIndexDefinition

Adobe

MAKE IT AN EXPERIENCE