Search Service Overview

» Allows running various queries:
   – Full-text
   – Field ranges
   – Field match
   – Sort, Aggregations, Filters
   – Geo-Spatial coordinates
   – Limits & Pagination

» Provides an API to do
   – queries
   – queries with cursor

» Requires membership in
   – `service.search.user`
   – the searched records’ ACL as Viewers
How Are Queries Executed?

» Record is indexed into Elasticsearch

» Query is performed against Elasticsearch Index

» JSON-style language to issue query
Query Syntax

```json
{
    "kind": "opendes:wks:master-data-Well:1.0.0",
    "query": "Text query",
    "offset": 0,
    "limit": 1000,
    "sort": {
        "fields": ["id"],
        "order": ["ASC"]
    },
    "queryAsOwner": true,
    "spatialFilter": {
        "field": "data.SpatialLocation.Wgs84Coordinates",
        "byDistance": {
            "point": {
                "longitude": 5.1580810546875,
                "latitude": 52.859180945520826
            },
            "distance": 10000
        }
    },
    "trackTotalCount": true,
    "returnedFields": ["id", "kind"],
    "aggregatedBy": "kind"
}
```
Query by Kind

// 1. look for all entities of given kind
{
    "kind": "opendes:wks:master-data--Well:1.0.0"
}

// 2. look for all wellbores of any version for wks source in opendes partition
{
    "kind": "opendes:wks:master-data--Wellbore:*"
}

// 3. search across all entity types and versions for wks source in opendes partition
{
    "kind": "opendes:wks:*:*"
}
Text Queries

// 1. full text search & look for in all fields
{
    "kind": "opendes:wks:master-data--Well:1.0.0",
    "query": "BIR"
}

// 2. field match search & wildcard
{
    "kind": "opendes:wks:master-data--Wellbore:*",
    "query": "data.WellID:(Well AND 100?)"
}

// 3. field match & exact text match
{
    "kind": "opendes:wks:master-data--Wellbore:*",
    "query": "id:"opendes:master-data--Wellbore:1000\"
}

Lucene Apache Query Syntax: https://lucene.apache.org/core/2_9_4/queryparsersyntax.html
// sort by field in descending order
{
    "kind": "opendes:wks:work-product-component--WellLog:*",
    "sort": {
        "field": ["data.BottomMeasuredDepth"],
        "order": ["DESC"]
    },
    "limit": 10
}
Aggregation

// count how many records are per kind for all kinds
{
    "kind": "*:*:*:*",
    "aggregateBy": "kind",
    "limit": 1000
}
Range Queries

// 1. get logs with bottom measured depth between 2000 and 3000
{
    "kind": "opendes:wks:work-product-component--WellLog:1.0.0",
    "query": "data.BottomMeasuredDepth:[2000 TO 3000]"
}

// 2. get entities with spud date between two dates
{
    "kind": "opendes:wks:*:1.0.0",
    "query": "(data.FacilityEvents.FacilityEventTypeID: "opendes:reference-data--FacilityEventType:SPUD:" ) AND (data.FacilityEvents.EffectiveDateTime:[1901-01-01 TO 2001-01-01])",
    "sort": {
        "field": ["data.FacilityEvents.EffectiveDateTime"],
        "order": ["DESC"]
    },
    "limit": 30
}
Geo-Spatial Queries

» By distance
» By bounding box
» By geo-polygon

// get wells within a given distance
{
  "kind": "opendes:wks:master-data--Well:1.0.0",
  "spatialFilter": {
    "field": "data.SpatialLocation.Wgs84Coordinates",
    "byDistance": {
      "point": {
        "longitude": 5.1580810546875,
        "latitude": 52.859180945520826
      },
      "distance": 10000
    }
  }
}
Query by Nested Array of Objects

Arrays of objects in Elasticsearch:
  – Flattened object
  – Nested objects

// single-level several conditions nested query
{
  "kind": "opendes:wks:work-product-component--wellboremarkerset:1.0.0",
  "query": "nested(data.Markers, (MarkerMeasuredDepth:(>1000) AND MarkerName:(Coppershale)))"
}
Pagination

» `query_with_cursor` endpoint

» *limit* and *offset* combination

```json
{
  "kind": "opendes:wks:master-data--Wellbore:*",
  "offset": 1,
  "limit": 100
}
```