Cloudera Search, which integrates Apache Solr with the Cloudera platform, is a powerful tool for performing searches over large datasets in Hadoop. Let's deep dive into its setup, configuration, and usage, including complex steps and examples.

Deep Dive into Cloudera Search

1. Overview of Cloudera Search with Apache Solr

- **Integration with Hadoop**: Cloudera Search leverages Apache Solr's scalability and full-text search capabilities, making it well-suited for searching through large volumes of data stored in Hadoop.
- **Near Real-Time Indexing**: With features like near real-time indexing, it offers quick search responses, even on vast datasets.

2. Pre-requisites

- Ensure you have a working Hadoop cluster managed by Cloudera Manager.
- Confirm that all nodes meet the hardware and software requirements for running Solr.

3. Installation via Cloudera Manager

- **Install Solr Service**: Use Cloudera Manager to add the Solr service to your cluster.
- **Configuration**: During installation, configure Solr with appropriate settings like memory allocation, number of nodes, etc.

4. Solr Collection and Schema Setup

- **Creating a Collection**: A collection in Solr is a logical index across Solr nodes.

  ```
  solrctl instancedir --generate $HOME/solr_configs
  solrctl instancedir --create collection1_configs $HOME/solr_configs
  solrctl collection --create collection1 -s 2 -c collection1_configs
  ```

- **Schema Configuration**: Define a schema for your data. Solr uses a schema to define the fields and types of data that can be indexed.

5. Data Ingestion and Indexing

- **Integrating with HDFS**: Configure Solr to index data stored in HDFS.
- **Indexing with MapReduce**: You can use MapReduce jobs to index large datasets into Solr.
  - Example (indexing a text file):

    ```
    hadoop jar $SOLR_HOME/example/solr/hadoop-map-reduce/solr-map-reduce-1.0.jar \
    -D 'mapred.child.java.opts=-Xmx500m' \
    -files $HOME/solr_configs \
    ```

```
-libjars $SOLR_HOME/dist/solr-solrj-*.jar,$SOLR_HOME/dist/solr-core-*.jar \
--morphline-file readAvroContainer.conf \
--output-dir hdfs:///solr_output \
--verbose \
--go-live
```

6. **Querying Data in Solr**
   - **Solr Queries**: Use Solr's query syntax to search through indexed data.
     - Example:
       `curl "http://[Solr_Host]:8983/solr/collection1/select?q=field:value&wt=json&indent=`
   - **Integration with Hue**: Cloudera Search can be accessed via Hue for a user-friendly querying interface.

7. **Performance Tuning and Scaling**
   - **Sharding and Replication**: Set up sharding and replication for scalability and high availability.
   - **Memory and Resource Management**: Tune memory settings for Solr instances for optimal performance.

8. **Security Configuration**
   - **Kerberos Authentication**: Integrate with Kerberos for secure access to Solr.
   - **Authorization**: Use Cloudera Manager to configure authorization rules for accessing the Solr service.

9. **Monitoring and Maintenance**
   - **Solr Metrics**: Monitor Solr performance metrics through Cloudera Manager.
   - **Regular Index Optimization**: Perform regular maintenance like index optimization and garbage collection.

Example: Advanced Search Scenario

Let's say you need to index and search a large dataset of web server logs stored in HDFS. The process would involve:

1. **Schema Design**: Define a schema corresponding to the log data structure.
2. **Data Indexing**: Use MapReduce to index the logs into Solr.
3. **Complex Queries**: Perform complex queries to analyze traffic patterns, error rates, or specific user activities.