IN THE UNITED STATES DISTRICT COURT
FOR THE WESTERN DISTRICT OF TEXAS
AUSTIN DIVISION

R2 Solutions LLC, Plaintiff, v. Cloudera, Inc., Defendant.	Civil Action No. 1:23-cv-1205 Jury Trial Demanded

COMPLAINT FOR PATENT INFRINGEMENT

Plaintiff R2 Solutions LLC files this Complaint against Cloudera, Inc. for infringement of U.S. Patent No. 8,190,610 ("the '610 patent"). The '610 patent is referred to as the "patent-in-suit."

THE PARTIES

1. Plaintiff R2 Solutions LLC ("R2") is a Texas limited liability company located in Frisco, Texas.

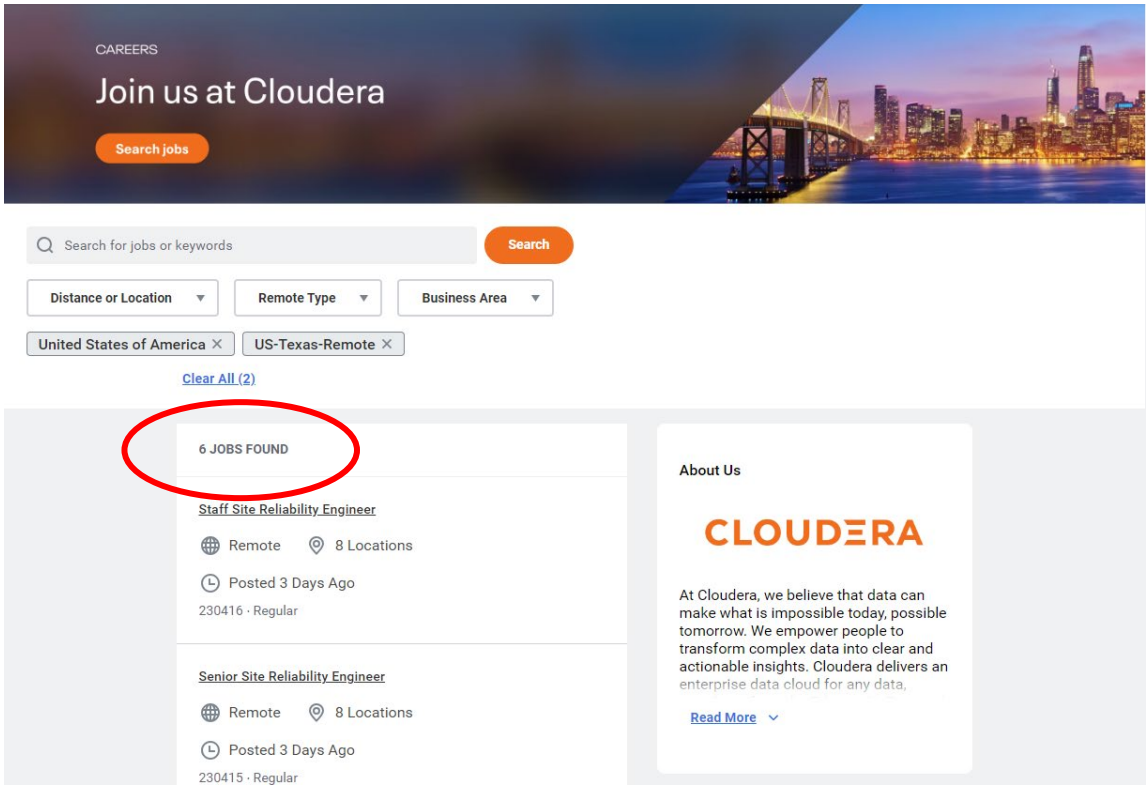
2. Defendant Cloudera, Inc. ("Cloudera") is a Delaware corporation headquartered at 5470 Great America Pkwy, Santa Clara, CA 95054 and having a regular and established place of business in this District at 515 Congress Ave, Suite 1300, Austin, TX 78701. Cloudera may be served with process through its registered agent, Corporation Service Company d/b/a CSC – Lawyers Incorporating Service Company, at 211 E. 7th Street, Suite 620, Austin, TX 78701.

JURISDICTION AND VENUE

3. This action arises under the patent laws of the United States, 35 U.S.C. § 101, *et seq.* This Court's jurisdiction over this action is proper under the above statutes, including 35

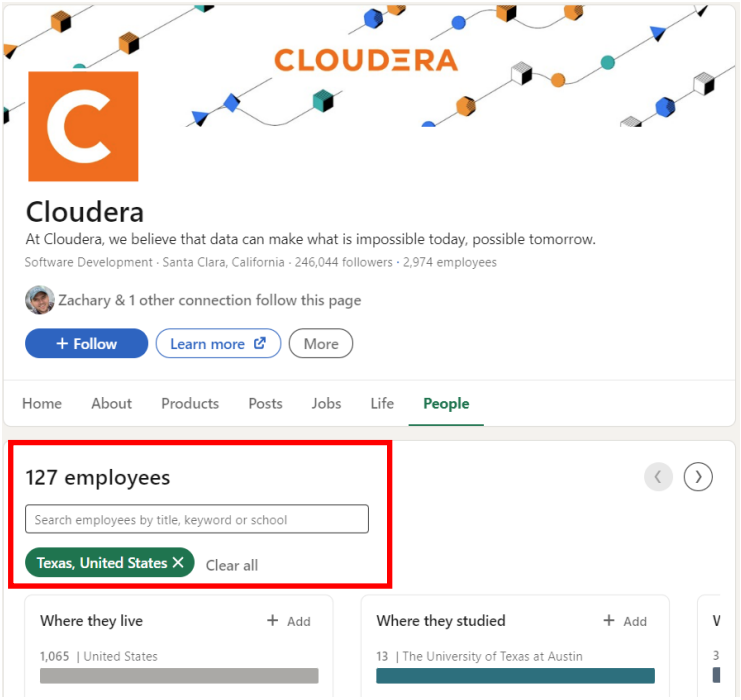
U.S.C. § 271, *et seq*., 28 U.S.C. § 1331 (federal question jurisdiction), and 28 U.S.C. § 1338 (jurisdiction over patent actions).

4. This Court has personal jurisdiction over Cloudera because, among other things, Cloudera does business in this State by, among other things, "recruit[ing] Texas residents, directly or through an intermediary located in this State, for employment inside or outside this State." TEX. CIV. PRAC. & REM. CODE § 17.042(3). For instance, Cloudera has multiple job openings in Texas as of October 2, 2023:[1]



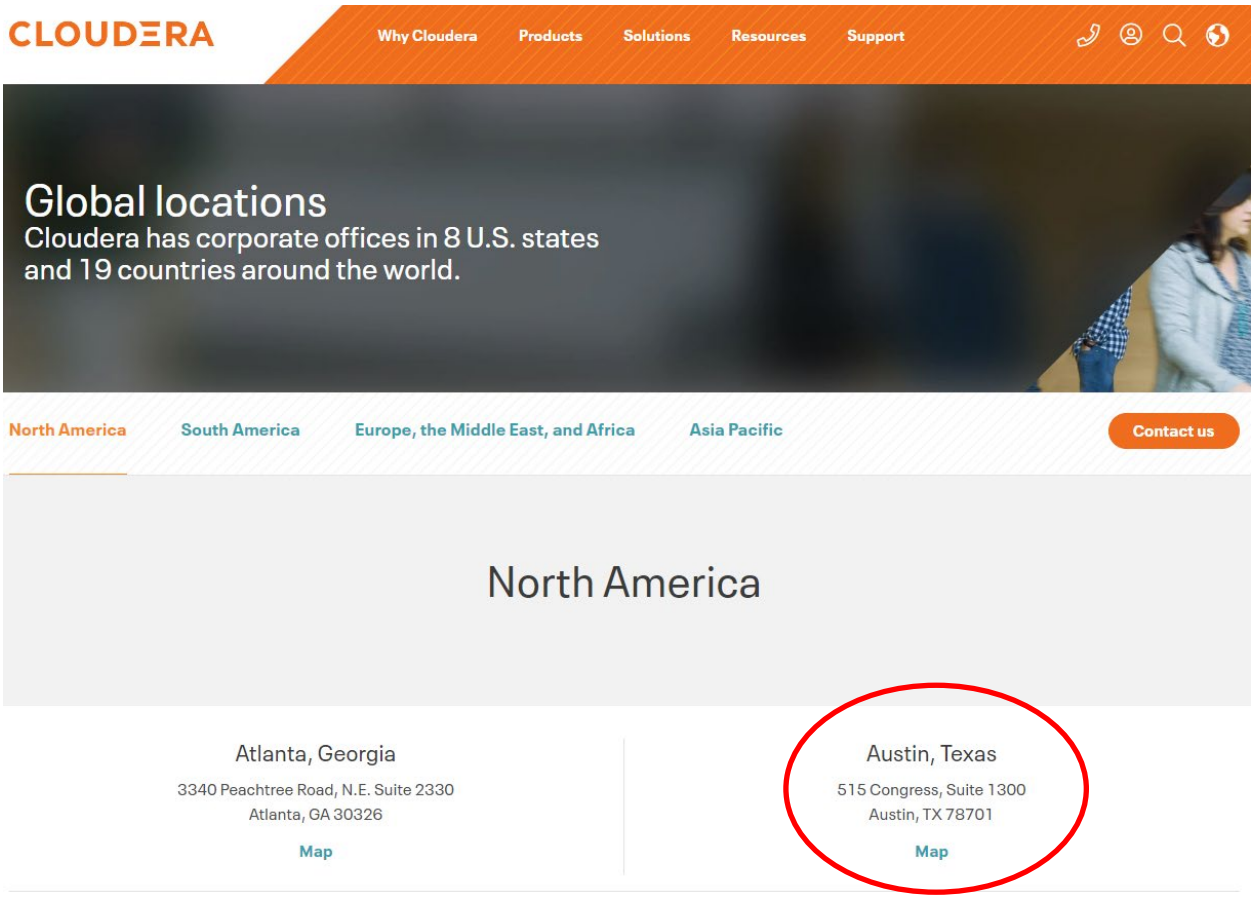
[1] https://cloudera.wd5.myworkdayjobs.com/External_Career?locationCountry=bc33aa3152ec42d4995f4791a106ed09&locations=099bd8052f77105bfed69a9cf552387f; *see also* https://www.linkedin.com/jobs/search/?currentJobId=3670951181&f_C=229433&geoId=92000000&keywords=texas.

5. And according to its LinkedIn page, Cloudera has 127 employees in its Austin office (as of October 2, 2023):[2]



6. Further, this Court has personal jurisdiction over Cloudera because it has engaged, and continues to engage, in continuous, systematic, and substantial activities within this State, including the substantial marketing and sale of products and services within this State and this District. Indeed, this Court has personal jurisdiction over Cloudera because it has committed acts giving rise to R2's claims for patent infringement within and directed to this District, has derived substantial revenue from its goods and services provided to individuals and entities in this State and this District, and maintains regular and established places of business in this District, including at least its brick-and-mortar location in Austin:[3]

[2] https://www.linkedin.com/company/cloudera/people/?facetGeoRegion=102748797.
[3] https://www.cloudera.com/about/locations.html.



7. Relative to patent infringement, Cloudera has committed and continues to commit acts in violation of 35 U.S.C. § 271, and has made, used, offered for sale, and/or sold infringing products, systems, and/or services in this State, including this District, and has otherwise engaged in infringing conduct within and directed at, or from, this District. Such infringing products, systems, and/or services (collectively, the "Accused Instrumentalities") include the Cloudera Data Platform (CDP) (including CDP Public Cloud, CDP Private Cloud, and CDP One), Cloudera Distributed Hadoop (CDH), Cloudera Enterprise (including at least the Data Science and Engineering, Operational DB, Data Warehouse, and/or Enterprise Data Hub editions), Hortonworks Data Platform (HDP), and any other platform(s) offered or provided by Cloudera that utilize one or more of Apache Hadoop, Apache Hive, Apache Spark, Apache Impala, Apache Flink, Apache Kafka, and Apache Phoenix.

8. Cloudera's infringing activities have caused harm to R2 in this District. Cloudera offers to sell, and sells, the Accused Instrumentalities to customers within this District, and on information and belief, Cloudera and/or its customers use the Accused Instrumentalities in this District in an infringing manner. For example, Cloudera and/or its customers (induced by Cloudera) implement and exert control over the Accused Instrumentalities via multi-cloud and on-premises solutions that utilize computers and/or servers located in this District. Outputs from such methods and systems are generated by and/or delivered to devices implementing the Accused Instrumentalities in this District. Cloudera's customers in this District obtain data analytics facilitated by the Accused Instrumentalities, whether via Cloudera's implementation of the Accused Instrumentalities on their behalf, or via their use of the Accused Instrumentalities provided to them by Cloudera. These are purposeful acts and transactions in this State and this District such that Cloudera reasonably should know and expect that it could be haled into this Court.

9. Venue is proper in this District under 28 U.S.C. §§ 1391 and 1400(b) because Cloudera has a regular and established place of business in Austin, which is in this District. Venue is further proper in this District because Cloudera has directly infringed and/or induced the infringement of others, including its customers, in this District. As set out above, Cloudera has at least offered for sale and sold the Accused Instrumentalities in this District and has used the Accused Instrumentalities in an infringing manner in this District. In addition, Cloudera's customers have used and continue to use the Accused Instrumentalities in an infringing manner in this District. These infringements were, and continue to be, induced by Cloudera (as set out further below).

BACKGROUND

10. The patent-in-suit was filed by Yahoo! Inc. ("Yahoo!") in 2006. At the time, Yahoo! was a leading Internet communications, commerce, and media company. Yahoo! invested billions of dollars in research and development over this period, filing hundreds of patent applications each year to cover the innovative computing technologies emerging from its expansive research and development efforts.

11. Yahoo! began as a directory of websites that two Stanford graduate students developed as a hobby. The name "Yahoo" stands for "Yet Another Hierarchical Officious Oracle," a nod to how the original Yahoo! database was arranged hierarchically in layers of subcategories. From this initial database, Yahoo! would develop and promulgate numerous advancements in the field of data storage and recall.

12. For example, in 1995, Yahoo! introduced Yahoo! Search. This software allowed users to search the Yahoo! directory, making it the first popular online directory search engine. This positioned Yahoo! as the launching point for most users of the World Wide Web. By 1998, Yahoo! had the largest audience of any website or online service. In the early 2000s, Yahoo! continued to develop its suite of technologies in the web search and database industries. The patent-in-suit relates to innovations during this period associated with data analytics.

THE PATENT-IN-SUIT

13. The '610 patent is entitled, "MapReduce for Distributed Database Processing." The '610 patent lawfully issued on May 29, 2012, and stems from U.S. Patent Application No. 11/539,090, which was filed on October 5, 2006. A copy of the '610 patent is attached hereto as Ex. 1.

14. R2 Solutions is the owner of the patent-in-suit with all substantial rights, including the exclusive right to enforce, sue, and recover damages for past and future infringements.

15. The claims of the patent-in-suit are directed to patent-eligible subject matter under 35 U.S.C. § 101. They are not directed to abstract ideas, and the technologies covered by the claims consist of ordered combinations of features and functions that, at the time of invention, were not, alone or in combination, well-understood, routine, or conventional.

16. Indeed, the specification of the patent-in-suit discloses shortcomings in the prior art and then explains, in detail, the technical way the claimed inventions resolve or overcome those shortcomings. For example, the specification explains that "conventional MapReduce implementations do not have facility to efficiently process data from heterogeneous sources" and that "it is impractical to perform joins over two relational tables that have different schemas." '610 patent at 3:9-20. To solve these problems, the '610 patent provides a clear technological improvement to existing MapReduce systems by describing and implementing a novel MapReduce architecture where mapping and reducing functions can be applied to data from heterogeneous data sources (i.e., data sources having different schema) to accomplish the merger of heterogeneous data based on a key in common between or among the heterogeneous data. For example, the '610 patent explains how implementation of, e.g., "data groups" realizes these improvements:

> In general, partitioning the data sets into data groups enables a mechanism to associate (group) identifiers with data sets, map functions and iterators (useable within reduce functions to access intermediate data) and, also, to produce output data sets with (group) identifiers. It is noted that the output group identifiers may differ from the input/intermediate group identifiers.

'610 patent at 3:58-64.

17. The technological advantages of a "data group"-centric system are shown to "enhance[] the utility of the MapReduce programming methodology." '610 patent at 1:32-33. As the specification explains:

> [T]he MapReduce concept may be utilized to carry out map processing independently on two or more related datasets (e.g., related by being characterized by a common key) even when the related data sets are heterogeneous with respect to each other, such as data tables organized according to different schema. The intermediate results of the map processing (key/value pairs) for a particular key can be processed together in a single reduce function by applying a different iterator to intermediate values for each group. In this way, operations on the two or more related datasets may be carried out more efficiently or in a way not even possible with the conventional MapReduce architecture.

Id. at 8:47-58.

18. Such a solution is embodied, for example, in Claim 1 of the '610 patent:

> A method of processing data of a data set over a distributed system, wherein the data set comprises a ***plurality of data groups***, the method comprising:
>
> partitioning the data of each one of the data groups into a plurality of data partitions that each have a plurality of key-value pairs and ***providing each data partition to a selected one of a plurality of mapping functions*** that are each user-configurable to independently output a plurality of lists of values for each of a set of keys found in such map function's corresponding data partition to form corresponding ***intermediate data for that data group and identifiable to that data group***, wherein ***the data of a first data group has a different schema than the data of a second data group*** and ***the data of the first data group is mapped differently than the data of the second data group*** so that different lists of values are output for the corresponding different intermediate data, ***wherein the different schema and corresponding different intermediate data have a key in common***; and

reducing the intermediate data for the data groups to at least one output data group, including ***processing the intermediate data for each data group in a manner that is defined to correspond to that data group***, so as to result in a ***merging of the corresponding different intermediate data based on the key in common***,

wherein the mapping and reducing operations are performed by a distributed system.

(emphasis added).

19. The concept of "data groups" as found in Claim 1 of the '610 patent in the context of MapReduce attains a novel and technological improvement in computer capabilities. For example, employing "data groups" allows a diverse data set to be fed to collections of mapping and reducing functions within the same MapReduce architecture to ultimately be joined and/or merged in spite of the diversity. Per Claim 1, the improved MapReduce architecture in the reducing phase is able to selectively employ specialized processing based on the "data group" from which the data being reduced originated, and this specialized processing enables the MapReduce architecture in the reducing phase to accomplish the merger of intermediate data hailing from different data groups.

20. The inventions described and claimed in the '610 patent improve the speed, efficiency, effectiveness, and functionality of computer systems. Moreover, the inventions provide an improvement in computer functionality rather than improvement in performance of an economic task or other tasks for which a computer is used merely as a tool. The '610 patent itself states that the claimed inventions "enhance[] the utility of the MapReduce programming methodology." '610 patent at Abstract, 1:31-33, 1:66-2:2. The '610 patent specification goes on to explain that "[t]he intermediate results of the map processing (key/value pairs) for a particular key can be processed together in a single reduce function by applying a different iterator to

intermediate values for each group." *Id.* at Abstract, 1:37-39, 2:4-8. And the specification discusses the use of multiple processors to perform processing functions in parallel. *See id.* As a result, computer functionality is improved. *Id.* at 1:42-44.

21. Additionally, the claimed inventions provide for more dynamic, customizable, and efficient processing of large sets of data. S*ee, e.g.*, '610 patent at 2:58-61, 4:18-22. The inventions provide optimization of such processing, which increases efficiency and reduces processor execution time. For example, the specification describes a combiner function that "helps reduce the network traffic and speed up the total execution time." '610 patent at 3:1-8. The specification also discusses the use of configurable settings to reduce processing overhead. *See, e.g.*, *id.* at 4:60-62, 5:33-39.

22. In essence, the patent-in-suit relates to novel and non-obvious inventions in the fields of data analytics and database structures.

DEFENDANT'S PRE-SUIT KNOWLEDGE OF ITS INFRINGEMENT

23. Prior to the filing of this Complaint, Cloudera was notified on numerous occasions of the '610 patent and the R2 portfolio to which the '610 patent belongs.

24. On March 2, 2021, R2 filed suit against JPMorgan Chase & Co. (the "JPM litigation") alleging infringement of the '610 patent.

25. On July 26, 2021, Craig Yudell, President of R2, sent a letter to David Howard, Cloudera's Chief Legal Officer, offering an opportunity to negotiate a license to the portfolio that includes the patent-in-suit. The letter explained that R2's portfolio originated from Yahoo! and that it includes patents covering a variety of technologies relevant to Cloudera. The letter further explained that R2 had asserted its patent rights against multiple companies, including in the JPM litigation.

26. On September 27, 2021, Mr. Yudell sent Mr. Howard another letter restating the information from the July 26 letter and, again, offering an opportunity to open negotiations. The September 27 letter further explained that since the July 26 letter, R2 had licensed many companies through negotiated deals and had also resolved several lawsuits. The patent-in-suit was asserted in these lawsuits.

27. Cloudera ignored these attempts to open a licensing dialogue.

28. On February 1, 2022, R2 served Cloudera with a subpoena in connection with the JPM litigation. The subpoena specifically identified the ’610 patent and sought materials and testimony regarding Cloudera’s systems and products that are now accused in this lawsuit.

29. On November 29, 2021, R2 filed suit against FedEx Corporate Services, Inc. (the “FedEx litigation”) alleging infringement of the ’610 patent.

30. On September 1, 2022, FedEx counsel filed a declaration with the Eastern District of Texas claiming that Cloudera possessed and controlled source code related to FedEx’s data analytics system.

31. On September 9, 2022, R2 served Cloudera with a subpoena in connection with the FedEx litigation. The subpoena specifically identified the ’610 patent and sought materials and testimony regarding Cloudera’s systems and products that are now accused in this lawsuit.

32. On information and belief, Cloudera has had knowledge of the ’610 patent and its infringements since shortly after March 2, 2021, when R2 filed suit in the JPM litigation. In the alternative, Cloudera has had knowledge of the ’610 patent since receiving the letter from Mr. Yudell on July 26, 2021. At the very least, Cloudera has had knowledge of the ’610 patent since being served with a subpoena in connection with the JPM litigation on February 1, 2022.

COUNT I
INFRINGEMENT OF U.S. PATENT NO. 8,190,610

33. This cause of action arises under the patent laws of the United States, and in particular, 35 U.S.C. §§ 271, *et seq*.

34. R2 Solutions is the owner of the ’610 patent with all substantial rights to the ’610 patent, including the exclusive right to enforce, sue, and recover damages for past and future infringements.

35. The ’610 patent is valid and enforceable and was duly issued in full compliance with Title 35 of the United States Code.

Direct Infringement (35 U.S.C. § 271(a))

36. Cloudera has directly infringed, and continues to directly infringe, one or more claims of the ’610 patent in this District and elsewhere in Texas and the United States.

37. To this end, Cloudera has infringed and continues to infringe, either by itself or via an agent, at least claims 1-32 of the ’610 patent by, among other things, making, offering to sell, selling, and/or using the Accused Instrumentalities.

38. Attached hereto as Ex. 2, and incorporated herein by reference, is representative claim charting detailing how Cloudera infringes the ’610 patent.

39. Cloudera is liable for its infringements of the ’610 patent pursuant to 35 U.S.C. § 271.

Indirect Infringement (Inducement – 35 U.S.C. § 271(b))

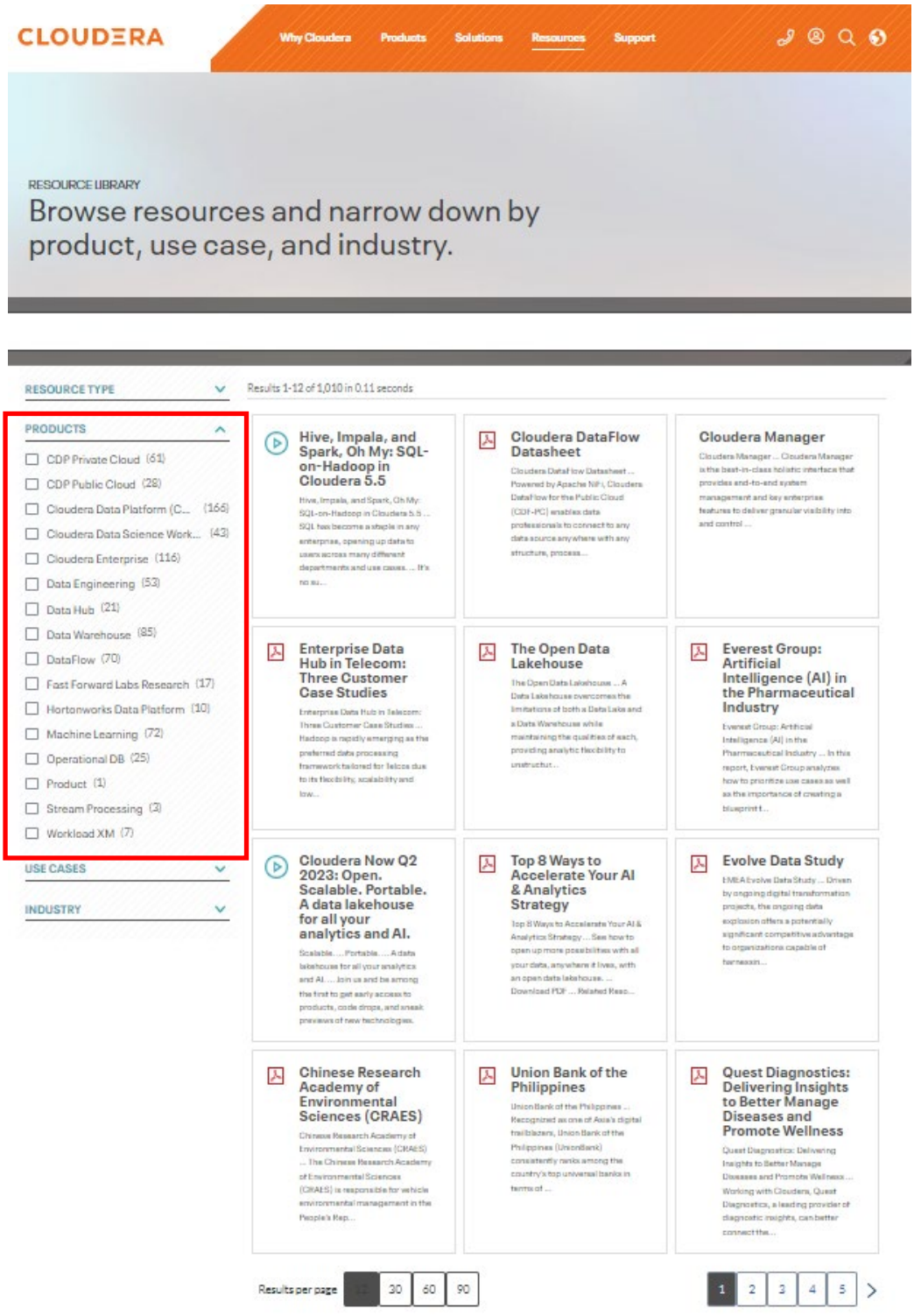
40. In addition and/or in the alternative to its direct infringements, Cloudera has indirectly infringed and continues to indirectly infringe one or more claims of the ’610 patent by inducing direct infringement by its customers and end users.

41. On information and belief, Cloudera has had knowledge of the ’610 patent since shortly after March 2, 2021, when R2 filed suit in the JPM litigation. In the alternative, Cloudera has had knowledge of the ’610 patent since receiving the letter from Mr. Yudell on July 26, 2021. At the very least, Cloudera has had knowledge of the ’610 patent since being served with a subpoena in connection with the JPM litigation on February 1, 2022.

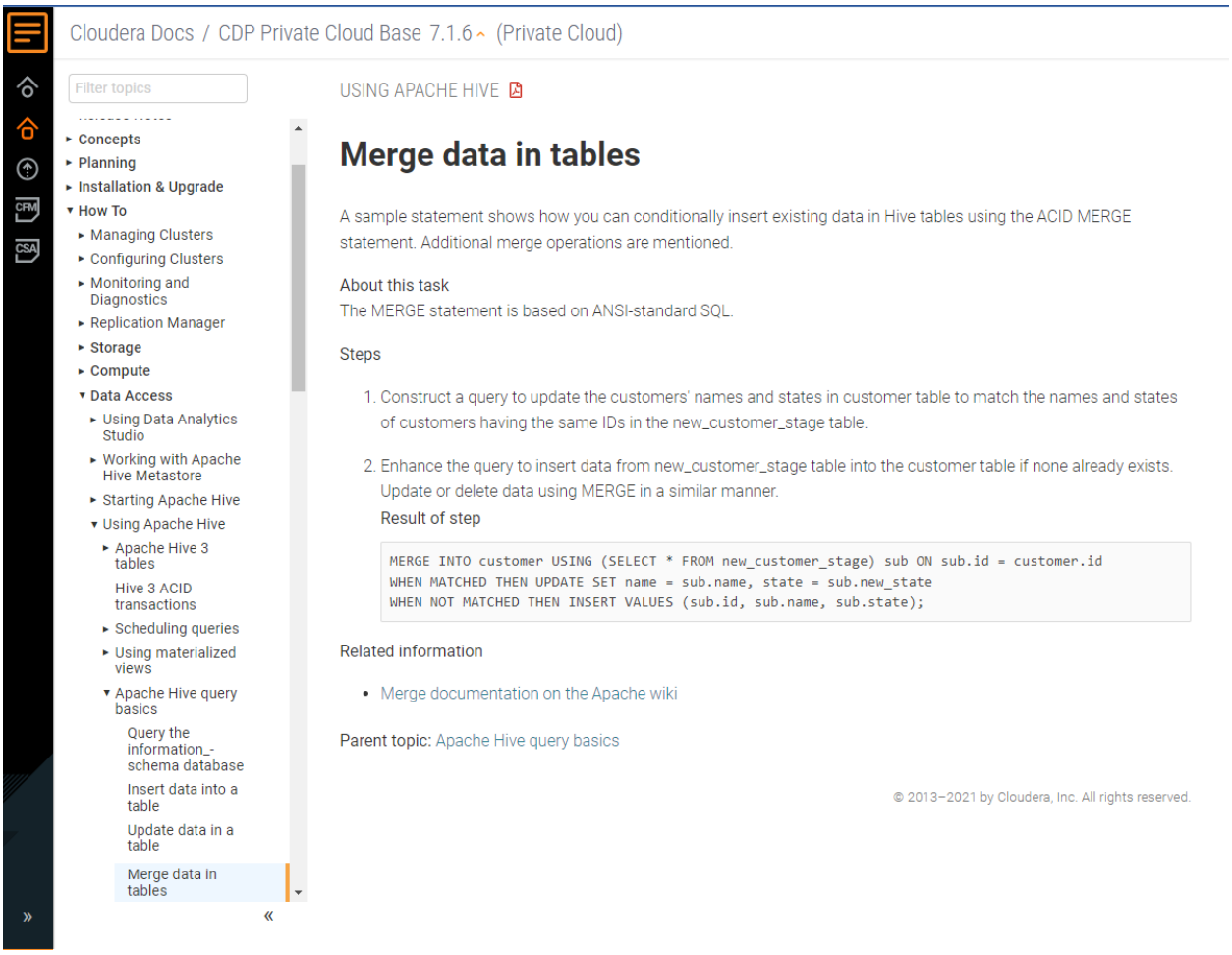
42. Despite having knowledge of the ’610 patent and knowledge of its scope, Cloudera has specifically intended, and continues to specifically intend, for persons (such as Cloudera’s customers and end users) to access, exercise control over, benefit from, use, and/or otherwise interact with the Accused Instrumentalities in ways that infringe the ’610 patent, including at least claims 1-32. Indeed, Cloudera knew or should have known that its actions have induced, and continue to induce, such infringements.

43. Cloudera instructs and encourages customers and end users to use the Accused Instrumentalities in ways that infringe the ’610 patent. For example, the Cloudera website includes a “Resources” page with explicit instructions on how to implement and operate each Accused Instrumentality:[4]

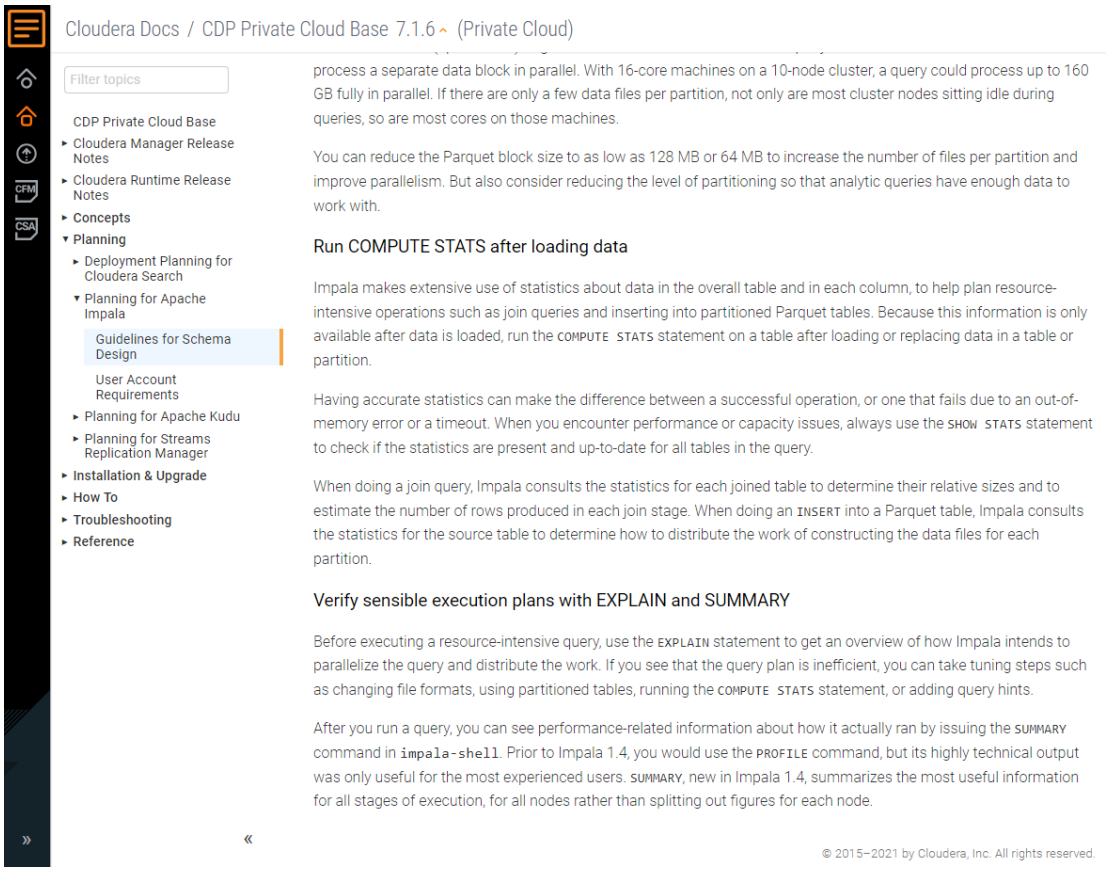
[4] https://www.cloudera.com/resources/resource-library.html#t=All&numberOfResults=12.


CLOUDERA
Why Cloudera
Products
Solutions
Resources
Support
RESOURCE LIBRARY
Browse resources and narrow down by product, use case, and industry.
RESOURCE TYPE
Results 1-12 of 1,010 in 0.11 seconds
PRODUCTS
CDP Private Cloud (61)
CDP Public Cloud (28)
Cloudera Data Platform (C... (166)
Cloudera Data Science Work... (43)
Cloudera Enterprise (116)
Data Engineering (53)
Data Hub (21)
Data Warehouse (85)
DataFlow (70)
Fast Forward Labs Research (17)
Hortonworks Data Platform (10)
Machine Learning (72)
Operational DB (25)
Product (1)
Stream Processing (3)
Workload XM (7)
USE CASES
INDUSTRY
Hive, Impala, and Spark, Oh My: SQL-on-Hadoop in Cloudera 5.5
Cloudera DataFlow Datasheet
Cloudera Manager
Enterprise Data Hub in Telecom: Three Customer Case Studies
The Open Data Lakehouse
Everest Group: Artificial Intelligence (AI) in the Pharmaceutical Industry
Cloudera Now Q2 2023: Open. Scalable. Portable. A data lakehouse for all your analytics and AI.
Top 8 Ways to Accelerate Your AI & Analytics Strategy
Evolve Data Study
Chinese Research Academy of Environmental Sciences (CRAES)
Union Bank of the Philippines
Quest Diagnostics: Delivering Insights to Better Manage Diseases and Promote Wellness
Results per page
30
60
90
1
2
3
4
5

44. Cloudera also maintains one or more "Cloudera Documentation" websites with explicit instructions on how to implement and operate each Accused Instrumentality in an infringing manner:[5]



[5] https://docs.cloudera.com/cdp-private-cloud-base/7.1.6/using-hiveql/topics/hive_merge_data_in_hive_tables.html; https://docs.cloudera.com/cdp-private-cloud-base/7.1.6/impala-planning/topics/impala-schema-design.html; https://docs.cloudera.com/documentation/enterprise/6/6.3/topics/hive_intro.html; https://docs.cloudera.com/documentation/enterprise/6/6.3/topics/impala_components.html; https://docs.cloudera.com/cdp-private-cloud-base/7.1.6/schema-registry-overview/topics/csp-schema_registry_overview.html; https://docs.cloudera.com/HDPDocuments/HDP2/HDP-2.1.3/bk_using-apache-hadoop/content/running_mapreduce_examples_on_yarn.html; https://docs.cloudera.com/documentation/enterprise/6/6.3/topics/spark.html.



Cloudera Docs / CDP Private Cloud Base 7.1.6 (Private Cloud)

Filter topics

- CDP Private Cloud Base
- Cloudera Manager Release Notes
- Cloudera Runtime Release Notes
- Concepts
- Planning
  - Deployment Planning for Cloudera Search
  - Planning for Apache Impala
    - Guidelines for Schema Design
    - User Account Requirements
  - Planning for Apache Kudu
  - Planning for Streams Replication Manager
- Installation & Upgrade
- How To
- Troubleshooting
- Reference

process a separate data block in parallel. With 16-core machines on a 10-node cluster, a query could process up to 160 GB fully in parallel. If there are only a few data files per partition, not only are most cluster nodes sitting idle during queries, so are most cores on those machines.

You can reduce the Parquet block size to as low as 128 MB or 64 MB to increase the number of files per partition and improve parallelism. But also consider reducing the level of partitioning so that analytic queries have enough data to work with.

Run COMPUTE STATS after loading data

Impala makes extensive use of statistics about data in the overall table and in each column, to help plan resource-intensive operations such as join queries and inserting into partitioned Parquet tables. Because this information is only available after data is loaded, run the `COMPUTE STATS` statement on a table after loading or replacing data in a table or partition.

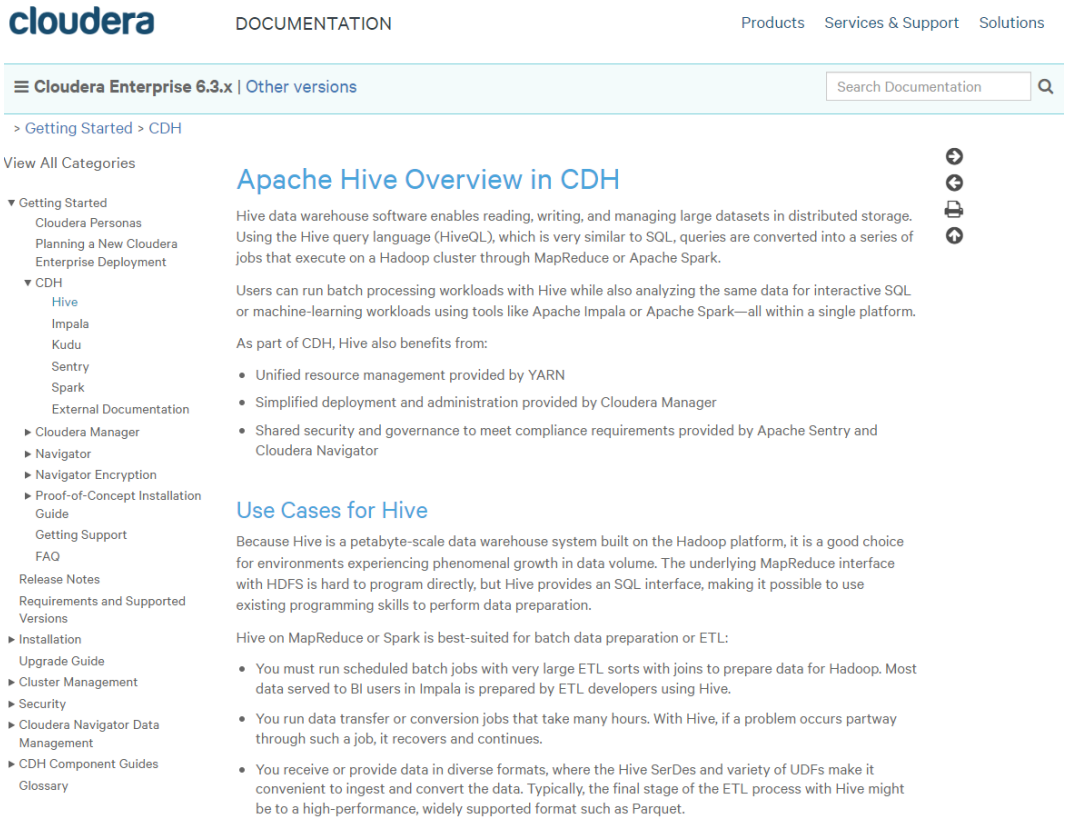
Having accurate statistics can make the difference between a successful operation, or one that fails due to an out-of-memory error or a timeout. When you encounter performance or capacity issues, always use the `SHOW STATS` statement to check if the statistics are present and up-to-date for all tables in the query.

When doing a join query, Impala consults the statistics for each joined table to determine their relative sizes and to estimate the number of rows produced in each join stage. When doing an `INSERT` into a Parquet table, Impala consults the statistics for the source table to determine how to distribute the work of constructing the data files for each partition.

Verify sensible execution plans with EXPLAIN and SUMMARY

Before executing a resource-intensive query, use the `EXPLAIN` statement to get an overview of how Impala intends to parallelize the query and distribute the work. If you see that the query plan is inefficient, you can take tuning steps such as changing file formats, using partitioned tables, running the `COMPUTE STATS` statement, or adding query hints.

After you run a query, you can see performance-related information about how it actually ran by issuing the `SUMMARY` command in `impala-shell`. Prior to Impala 1.4, you would use the `PROFILE` command, but its highly technical output was only useful for the most experienced users. `SUMMARY`, new in Impala 1.4, summarizes the most useful information for all stages of execution, for all nodes rather than splitting out figures for each node.



cloudera DOCUMENTATION Products Services & Support Solutions

Cloudera Enterprise 6.3.x | Other versions

Search Documentation

> Getting Started > CDH

View All Categories

- Getting Started
  - Cloudera Personas
  - Planning a New Cloudera Enterprise Deployment
  - CDH
    - Hive
    - Impala
    - Kudu
    - Sentry
    - Spark
    - External Documentation
  - Cloudera Manager
  - Navigator
  - Navigator Encryption
  - Proof-of-Concept Installation Guide
  - Getting Support
  - FAQ
- Release Notes
- Requirements and Supported Versions
- Installation
- Upgrade Guide
- Cluster Management
- Security
- Cloudera Navigator Data Management
- CDH Component Guides
- Glossary

Apache Hive Overview in CDH

Hive data warehouse software enables reading, writing, and managing large datasets in distributed storage. Using the Hive query language (HiveQL), which is very similar to SQL, queries are converted into a series of jobs that execute on a Hadoop cluster through MapReduce or Apache Spark.

Users can run batch processing workloads with Hive while also analyzing the same data for interactive SQL or machine-learning workloads using tools like Apache Impala or Apache Spark—all within a single platform.

As part of CDH, Hive also benefits from:

- Unified resource management provided by YARN
- Simplified deployment and administration provided by Cloudera Manager
- Shared security and governance to meet compliance requirements provided by Apache Sentry and Cloudera Navigator

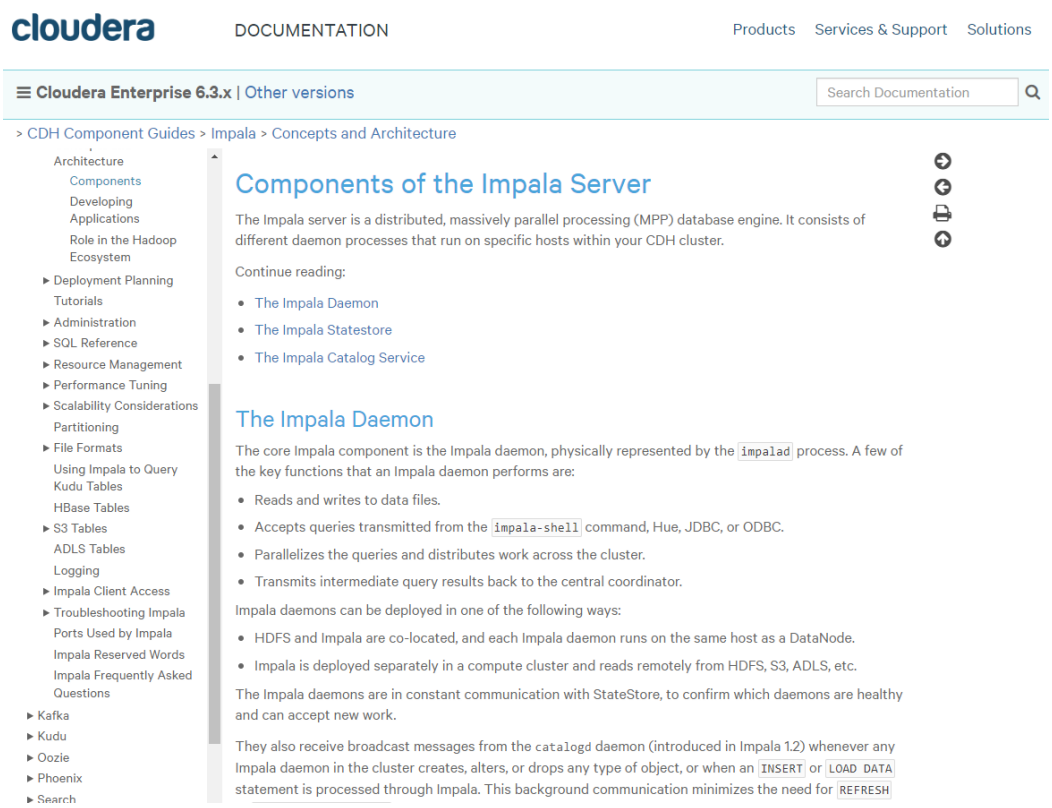
Use Cases for Hive

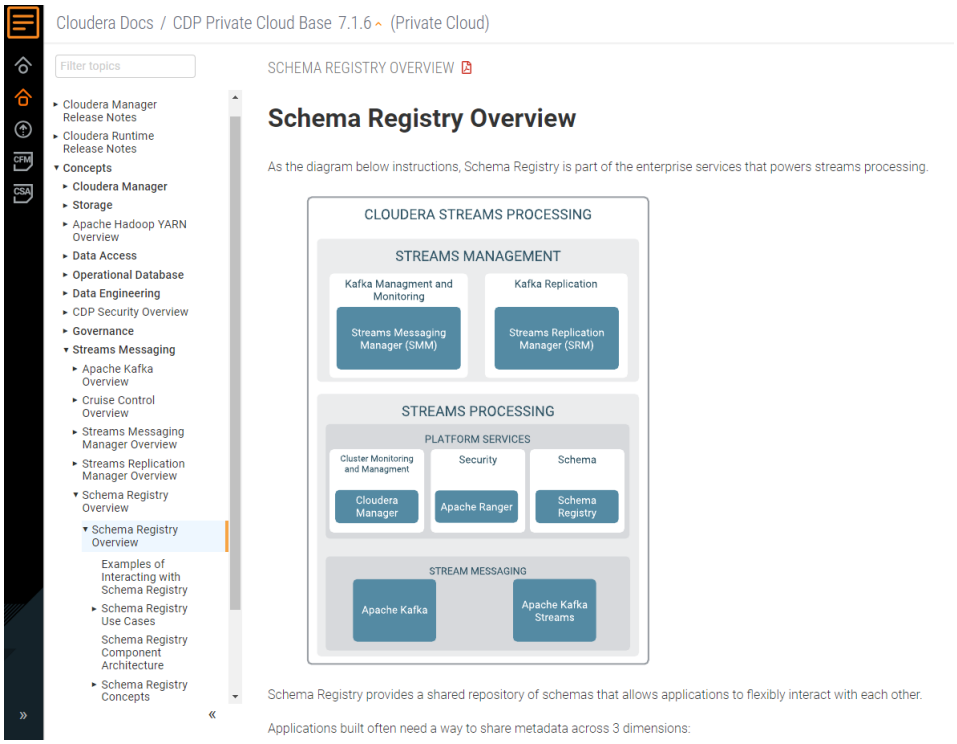
Because Hive is a petabyte-scale data warehouse system built on the Hadoop platform, it is a good choice for environments experiencing phenomenal growth in data volume. The underlying MapReduce interface with HDFS is hard to program directly, but Hive provides an SQL interface, making it possible to use existing programming skills to perform data preparation.

Hive on MapReduce or Spark is best-suited for batch data preparation or ETL:

- You must run scheduled batch jobs with very large ETL sorts with joins to prepare data for Hadoop. Most data served to BI users in Impala is prepared by ETL developers using Hive.
- You run data transfer or conversion jobs that take many hours. With Hive, if a problem occurs partway through such a job, it recovers and continues.
- You receive or provide data in diverse formats, where the Hive SerDes and variety of UDFs make it convenient to ingest and convert the data. Typically, the final stage of the ETL process with Hive might be to a high-performance, widely supported format such as Parquet.

.


cloudera
DOCUMENTATION
Products
Services & Support
Solutions
Cloudera Enterprise 6.3.x | Other versions
Search Documentation
> CDH Component Guides > Impala > Concepts and Architecture
Architecture
Components
Developing Applications
Role in the Hadoop Ecosystem
Deployment Planning
Tutorials
Administration
SQL Reference
Resource Management
Performance Tuning
Scalability Considerations
Partitioning
File Formats
Using Impala to Query Kudu Tables
HBase Tables
S3 Tables
ADLS Tables
Logging
Impala Client Access
Troubleshooting Impala
Ports Used by Impala
Impala Reserved Words
Impala Frequently Asked Questions
Kafka
Kudu
Oozie
Phoenix
Search
Components of the Impala Server
The Impala server is a distributed, massively parallel processing (MPP) database engine. It consists of different daemon processes that run on specific hosts within your CDH cluster.
Continue reading:
The Impala Daemon
The Impala Statestore
The Impala Catalog Service
The Impala Daemon
The core Impala component is the Impala daemon, physically represented by the impalad process. A few of the key functions that an Impala daemon performs are:
Reads and writes to data files.
Accepts queries transmitted from the impala-shell command, Hue, JDBC, or ODBC.
Parallelizes the queries and distributes work across the cluster.
Transmits intermediate query results back to the central coordinator.
Impala daemons can be deployed in one of the following ways:
HDFS and Impala are co-located, and each Impala daemon runs on the same host as a DataNode.
Impala is deployed separately in a compute cluster and reads remotely from HDFS, S3, ADLS, etc.
The Impala daemons are in constant communication with StateStore, to confirm which daemons are healthy and can accept new work.
They also receive broadcast messages from the catalogd daemon (introduced in Impala 1.2) whenever any Impala daemon in the cluster creates, alters, or drops any type of object, or when an INSERT or LOAD DATA statement is processed through Impala. This background communication minimizes the need for REFRESH


Cloudera Docs / CDP Private Cloud Base 7.1.6 (Private Cloud)
Filter topics
Cloudera Manager Release Notes
Cloudera Runtime Release Notes
Concepts
Cloudera Manager
Storage
Apache Hadoop YARN Overview
Data Access
Operational Database
Data Engineering
CDP Security Overview
Governance
Streams Messaging
Apache Kafka Overview
Cruise Control Overview
Streams Messaging Manager Overview
Streams Replication Manager Overview
Schema Registry Overview
Schema Registry Overview
Examples of Interacting with Schema Registry
Schema Registry Use Cases
Schema Registry Component Architecture
Schema Registry Concepts
SCHEMA REGISTRY OVERVIEW
Schema Registry Overview
As the diagram below instructions, Schema Registry is part of the enterprise services that powers streams processing.
CLOUDERA STREAMS PROCESSING
STREAMS MANAGEMENT
Kafka Managment and Monitoring
Kafka Replication
Streams Messaging Manager (SMM)
Streams Replication Manager (SRM)
STREAMS PROCESSING
PLATFORM SERVICES
Cluster Monitoring and Managment
Security
Schema
Cloudera Manager
Apache Ranger
Schema Registry
STREAM MESSAGING
Apache Kafka
Apache Kafka Streams
Schema Registry provides a shared repository of schemas that allows applications to flexibly interact with each other.
Applications built often need a way to share metadata across 3 dimensions:

Hortonworks

3.1.1. Running MapReduce Examples on Hadoop YARN

CONTENTS SEARCH

DOCS › HORTONWORKS DATA PLATFORM

- 1. Using Apache Hadoop
  - 1. Hadoop Common
  - 2. Using Hadoop HDFS
  - 3. Using Hadoop MapReduce on YARN
    - 3.1. Running MapReduce on Hadoop YARN
      - 3.1.1. Running MapReduce Examples on Hadoop YARN
      - 3.1.2. MapReduce Compatibility
      - 3.1.3. The MapReduce Application Master
      - 3.1.4. Calculating the Capacity of a Node
      - 3.1.5. Changes to the Shuffle Service
      - 3.1.6. Running Existing Hadoop Version 1 Applications on YARN
      - 3.1.7. Running Existing MapReduce Version 1 Code on YARN
      - 3.1.8. Uber Jobs (Technical Preview)
    - 3.2. MapReduce Version 2 Troubleshooting Guide
    - 3.3. YARN Components
    - 3.4. Using Hadoop YARN

3.1.1. Running MapReduce Examples on Hadoop YARN

The MapReduce examples are located in `hadoop-[VERSION]/share/hadoop/mapreduce`. Depending on where you installed Hadoop, this path may vary. For the purposes of this example let's define:

```
export YARN_EXAMPLES=$YARN_HOME/share/hadoop/mapreduce
```

`$YARN_HOME` should be defined as part of your installation. Also, the following examples have a version tag, in this case "2.1.0-beta." Your installation may have a different version tag.

The following sections provide some examples of Hadoop YARN programs and benchmarks.

Listing Available Examples

Using our `$YARN_HOME` environment variable, we can get a list of the available examples by running:

```
yarn jar $YARN_EXAMPLES/hadoop-mapreduce-examples-2.1.0-beta.jar
```

This command returns a list of the available examples:

```
An example program must be given as the first argument.
Valid program names are:
  aggregatewordcount: An Aggregate-based map/reduce program that counts
  the words in the input files.
  aggregatewordhist: An Aggregate-based map/reduce program that computes
  the histogram of the words in the input files.
  bbp: A map/reduce program that uses Bailey-Borwein-Plouffe to compute
  exact digits of Pi.
  dbcount: An example job that counts the pageview counts from a database.
  distbbp: A map/reduce program that uses a BBP-type formula to compute
  exact bits of Pi.
  grep: A map/reduce program that counts the matches of a regex in the
  input.
  join: A job that effects a join over sorted, equally partitioned
  datasets.
  multifilewc: A job that counts words from several files.
  pentomino: A map/reduce tile-laying program that finds solutions to
  pentomino problems.
  pi: A map/reduce program that estimates Pi using a quasi-Monte Carlo
  method.
  randomtextwriter: A map/reduce program that writes 10GB of random
  textual data per node.
  randomwriter: A map/reduce program that writes 10GB of random data per
  node.
  secondarysort: An example defining a secondary sort to the reduce.
  sort: A map/reduce program that sorts the data written by the random
  writer.
  sudoku: A sudoku solver.
  teragen: Generate data for the terasort.
  terasort: Run the terasort.
  teravalidate: Check the results of the terasort.
  wordcount: A map/reduce program that counts the words in the input
```

cloudera DOCUMENTATION Products Services & Support Solutions

Cloudera Enterprise 6.3.x | Other versions

Search Documentation

> CDH Component Guides

- Management
- CDH Component Guides
  - Crunch
  - Flume
  - HBase
  - Hive
  - Hue
  - Impala
  - Kafka
  - Kudu
  - Oozie
  - Phoenix
  - Search
  - Sentry
  - Spark
    - Running Your First Spark Application
    - Troubleshooting for Spark
    - Frequently Asked Questions about Apache Spark in CDH
    - Spark Application Overview
    - Developing Spark Applications
    - Running Spark Applications
    - Spark and Hadoop Integration
  - File Formats and Compression

Spark Guide

Note:
This page contains information related to Spark 2.x, which is included with CDH beginning with CDH 6. This information supercedes the documentation for the separately available parcel for CDS Powered By Apache Spark.

Apache Spark is a general framework for distributed computing that offers high performance for both batch and interactive processing. It exposes APIs for Java, Python, and Scala and consists of Spark core and several related projects.

You can run Spark applications locally or distributed across a cluster, either by using an interactive shell or by submitting an application. Running Spark applications interactively is commonly performed during the data-exploration phase and for ad hoc analysis.

To run applications distributed across a cluster, Spark requires a cluster manager. In CDH 6, Cloudera supports only the YARN cluster manager. When run on YARN, Spark application processes are managed by the YARN ResourceManager and NodeManager roles. Spark Standalone is no longer supported.

For detailed API information, see the Apache Spark project site.

Note: Although this document makes some references to the external Spark site, not all the features, components, recommendations, and so on are applicable to Spark when used on CDH. Always cross-check the Cloudera documentation before building a reliance on some aspect of Spark that might not be supported or recommended by Cloudera. In particular, see Apache Spark Known Issues for components and features to avoid.

The Apache Spark 2 service in CDH 6 consists of Spark core and several related projects:

Spark SQL

45. Other exemplary instructions and documentation that explain how to implement and operate the Accused Instrumentalities in an infringing manner are set out in Exhibit 2.

Damages

46. R2 has been damaged as a result of Cloudera's infringing conduct described in this Count. Cloudera is, thus, liable to R2 in an amount that adequately compensates it for Cloudera's infringements, which, by law, cannot be less than a reasonable royalty, together with interest and costs as fixed by this Court under 35 U.S.C. § 284.

47. Despite having knowledge of the '610 patent, and knowledge that it is potentially directly and/or indirectly infringing claims of the '610 patent, Cloudera has nevertheless continued its infringing conduct in an egregious manner. On information and belief, Cloudera knew of the '610 patent and its scope, yet continued to manufacture and sell infringing products. At the very least, Cloudera was willfully blind to the '610 patent and its application to the Accused Instrumentalities. For at least these reasons, Cloudera's infringing activities have been, and continue to be, willful, wanton, and deliberate in disregard of R2's rights with respect to the '610 patent, justifying enhanced damages under 35 U.S.C. § 284.

DEMAND FOR A JURY TRIAL

R2 demands a trial by jury on all issues triable of right by jury pursuant to Rule 38 of the Federal Rules of Civil Procedure.

PRAYER FOR RELIEF

R2 respectfully requests that this Court enter judgment in its favor and grant the following relief:

(i) Judgment and Order that Cloudera has directly and/or indirectly infringed one or more claims of the patent-in-suit;

(ii) Judgment and Order that Cloudera must pay R2 past and future damages under 35 U.S.C. § 284, including supplemental damages arising from any continuing, post-verdict infringement for the time between trial and entry of the final judgment, together with an accounting, as needed, as provided under 35 U.S.C. § 284;

(iii) Judgment and Order that Cloudera must pay R2 reasonable ongoing royalties on a go-forward basis after Final Judgment;

(iv) Judgment and Order that Cloudera's infringement of the '610 patent has been willful from the time that Cloudera became aware of the infringing nature of its products, and that the Court award treble damages pursuant to 35 U.S.C. § 284;

(v) Judgment and Order that Cloudera must pay R2 pre-judgment and post-judgment interest on the damages award;

(vi) Judgment and Order that Cloudera must pay R2's costs;

(vii) Judgment and Order that the Court find this case exceptional under the provisions of 35 U.S.C. § 285 and, accordingly, order Cloudera to pay R2's attorneys' fees; and

(viii) Such other and further relief as the Court may deem just and proper.

Dated: October 5, 2023

Respectfully submitted,

/s/ Edward R. Nelson III

EDWARD R. NELSON III
State Bar No. 00797142
ed@nelbum.com
BRENT N. BUMGARDNER
State Bar No. 00795272
brent@nelbum.com
CHRISTOPHER G. GRANAGHAN
State Bar No. 24078585
chris@nelbum.com
JOHN P. MURPHY

State Bar No. 24056024
murphy@nelbum.com
CARDER W. BROOKS
State Bar No. 24105536
carder@nelbum.com
NELSON BUMGARDNER CONROY PC
3131 West 7th Street, Suite 300
Fort Worth, Texas 76107
817.377.9111

COUNSEL FOR PLAINTIFF
R2 SOLUTIONS LLC