## What is Impala?

* Massive Parallel processing (MPP) database engine, developed by Cloudier
* Integrated into Hadoop stack on the same level as Map-Reduce, and not above it (as Hive and Pig)

|  |
| --- |
| HIVE  PIG  IMPALA  Map Reduce  HDFS |

## Why Impala?

* Today a lot of data live in HDFS
* It is not practical to move big data
* It is practical to bring engine to the data
* In the same time – Map Reduce is not must
* Impala process data in Hadoop cluster **without** using MapReduce

### Interactive SQL

* Typically 5-65x faster than Hive(observed up to 100x faster)
* Responses in seconds instead of minutes (sommeliers sub-second)

### Approx. ANSI-92 standard SQL queries with HiveQL

* compellable SQL interface for existing Hadoop/CDH applications
* Based on industry standard SQL

## Impala Architecture

* Impala is composed largely of impaled and impala state store.
* Impala is a process that functions as a distributed query engine. It designs a plan for queries and processes queries on data nodes in the Hadoop cluster. The impala state store process maintains metadata for the impalas executed on each data node. When the impala process is added or deleted in the cluster, metadata is updated through the impala state store process

Common Hive SQL and Interface Unified metadata

**HDFS** **DN**

**HBase**

**HDFS** **DN**

**HBase**

**HDFS** **DN**

**SQL App**

**ODBC**

**Hive** **Metastore**

**HDFS** **NN**

**State Store**

**Query Planner**

**Query Coordinator**

**Query Exec Engine**

**HBase**

**Query Planner**

**Query Coordinator**

**Query Exec Engine**

**Query Exec Engine**

**Query Planner**

**Query Coordinator**

Fully MPP

Distributed

## Query Execution Phases

* Client SQL arrives via ODBC/JDBC/Hue GUI/Shell
* Planner turns request into collection s of plan fragments
* Coordinator initiates executer on impalas local to data
* During Execution
* Intermediate results are streamed between executers
* Query Results are streamed back to client
* Subject to limitations imposed to blocking operators (top-n, aggregation)

## Query Exe Phase1

Step 1: Clients sends a query to any impala Daemon in the cluster

I can receive queries and prepare an execution plan with help from query planner

Now, I have Change this query into collections of plan fragment

|  |  |
| --- | --- |
| **Impala Daemon-0** | |
| Query Planner | |
| Query Coordinator | |
| Query Executer | |
| HDFS | HDASE |
| Hadoop Name Node | |

I am constantly watching you daemons

I am active and doing little work & can handle more load

I am active yet it has been long to have done some real work

Client Sends Query To Any Impala Daemon

STATE STORE

Don’t bother me, too busy with load

|  |  |
| --- | --- |
| **Impala Daemon-3** | |
| Query Planner | |
| Query Coordinator | |
| Query Executer | |
| HDFS | HDASE |
| Hadoop Name Node | |

|  |  |
| --- | --- |
| **Impala Daemon-2** | |
| Query Planner | |
| Query Coordinator | |
| Query Executer | |
| HDFS | HDASE |
| Hadoop Name Node | |

|  |  |
| --- | --- |
| **Impala Daemon-1** | |
| Query Planner | |
| Query Coordinator | |
| Query Executer | |
| HDFS | HDASE |
| Hadoop Name Node | |

## Query Exe Phase2

|  |  |
| --- | --- |
| **Impala Daemon-0** | |
| Query Planner | |
| Query Coordinator | |
| Query Executer | |
| HDFS | HDASE |
| Hadoop Name Node | |

It is my turn to get into the act now. I see ID-1 and ID-2 to be available to help me execute part of my query to get results

I am constantly watching you daemons

STATE STORE

|  |  |
| --- | --- |
| **Impala Daemon-1** | |
| Query Planner | |
| Query Coordinator | |
| Query Executer | |
| HDFS | HDASE |
| Hadoop Name Node | |

|  |  |
| --- | --- |
| **Impala Daemon-2** | |
| Query Planner | |
| Query Coordinator | |
| Query Executer | |
| HDFS | HDASE |
| Hadoop Name Node  Yes, I am almost done with my previous assignment and can take more now | |

Finally, something interesting to do has come from Daemon-0, will execute on HDFS

## HIVE – IMPALA

* Everything client-facing is shared with Hive:
* Metadata (table definitions)
* ODBC/JDBC drivers
* Hue GUI
* SQL syntax (HiveQL)
* Flexible file formats
* Machine pool

## Internal Improvements:

* Purpose-built query engine direct on HDFS and Hbase
* No JVM startup and no MapReduce
* In-memory data transfers
* Modern tech including special hardware instructors, runtime code generation, etc.
* Nalve distributed relational query engine

## Features not available in Impala but available in Hive

* Non-scalar data types such as maps, arrays, structs.
* Extensibility mechanisms such as TRANSFORM, custom file formats, or custom SerDes
* XML and JSON functions.
* Certain aggregate functions from
* Sampling
* Lateral views
* Multiple DISTINCT clauses per query, although impala includes some workarounds for this limitation
* ANALYZE TABLE(the impala equivalent is COMPUTE STATS)
* DESCRIBE COLUMN
* DESCRIBE DATABASE
* EXPORT TABLE, IMPORT TABLE, SHOW TABLE EXTENDED, SHOW INDEXES, SHOW COLUMNS