# WHAT’S IMPALA

# What’s Impala ?

- In-Memory, Distributed SQL query Engine ( No MapReduce )
- Native Backend Code ( C++)
- Distributed on HDFS data Nodes

# Goal of Impala

- A  general SQL engine for distributed system Supporting both OLTP and OLAP
- Interactive ( real-time ) queries.
- Built on top of HDFS and HBase
- Engine is written in C++, fast
- The database execution engine is like that of massively parallel processing ( MPP) database not using MapReduce

- **Interactive SQL**
  - Typically5-65x faster than Hive ( observed up to 100x faster)
  - Responses in seconds instead of minutes ( sometimes sub-seconds)

- **Approx. ANSI-92 Standard SQL queries with HiveSQL**
  - Compatible SQL interface for existing Hadoop/CDH application
  - Based on industry standard SQL

- **Natively on Hadoop/Hbase  storage and metadata**
  - Flexibility, scale and cost advances of Hadoop
  - No duplication/synchronization of data and metadata
  - Local processing to avoid network bottleneck

- **Separate runtime from MapReduce**
  - MapReduce is designed and great for batch
  - Impala is purpose-built for low-latnency SQL queries on Hadoop

# Why Impala ?

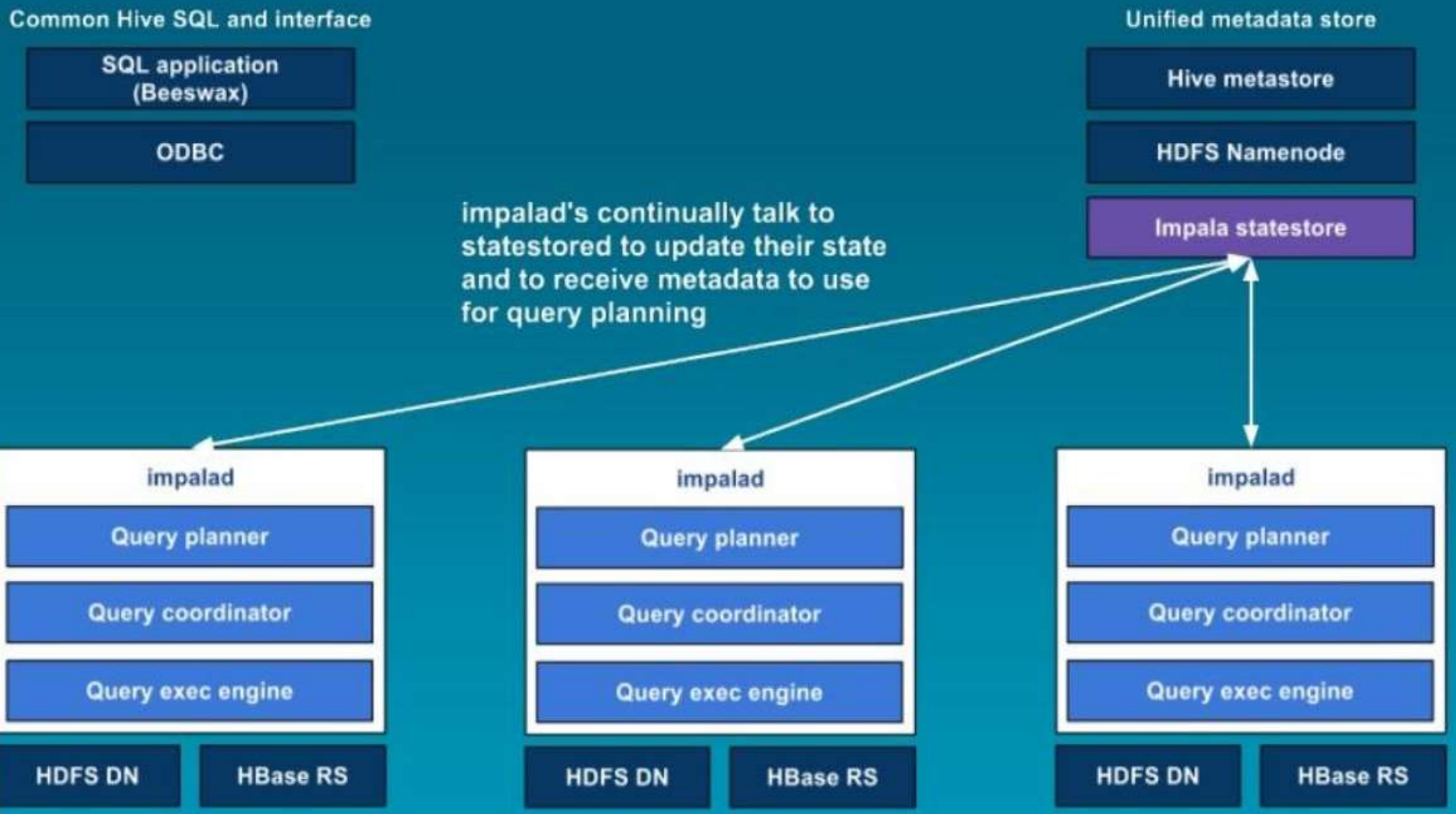
FAST !

# Why HDFS ?

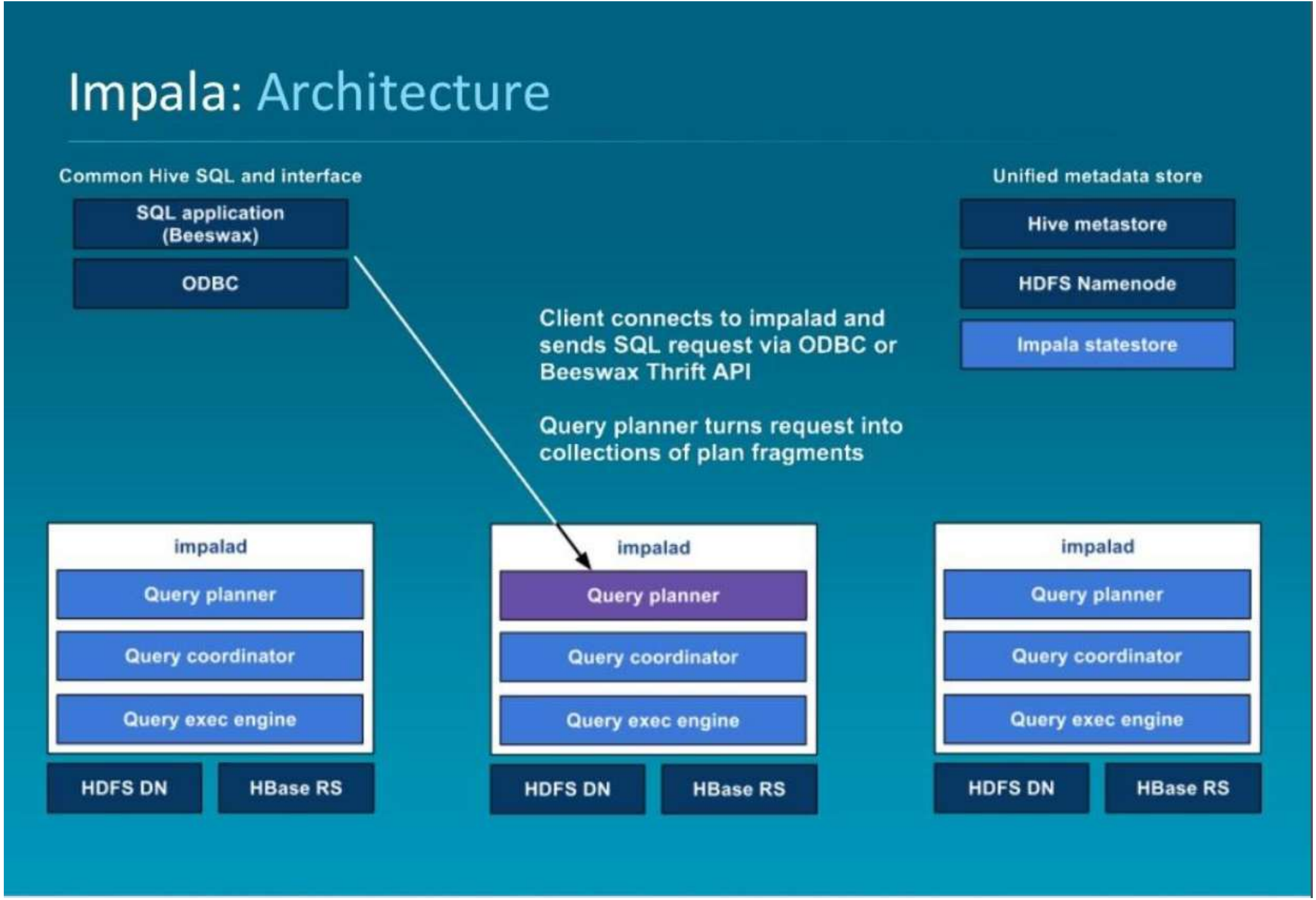
- ❑ Low Cost
- ❑ Reliability
- ❑ Easy to scale out

# ARCHITECTURE

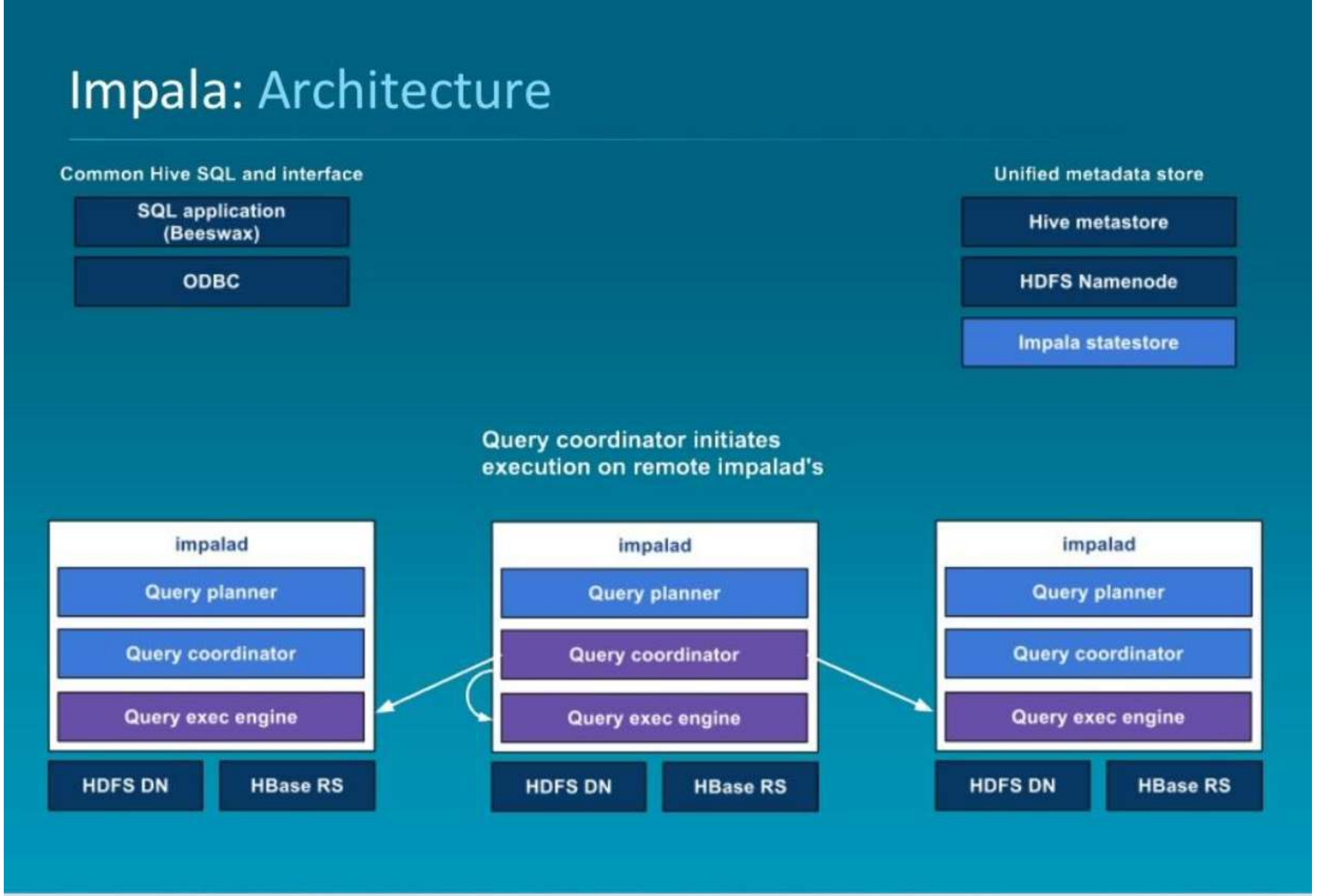
# Impala: Architecture



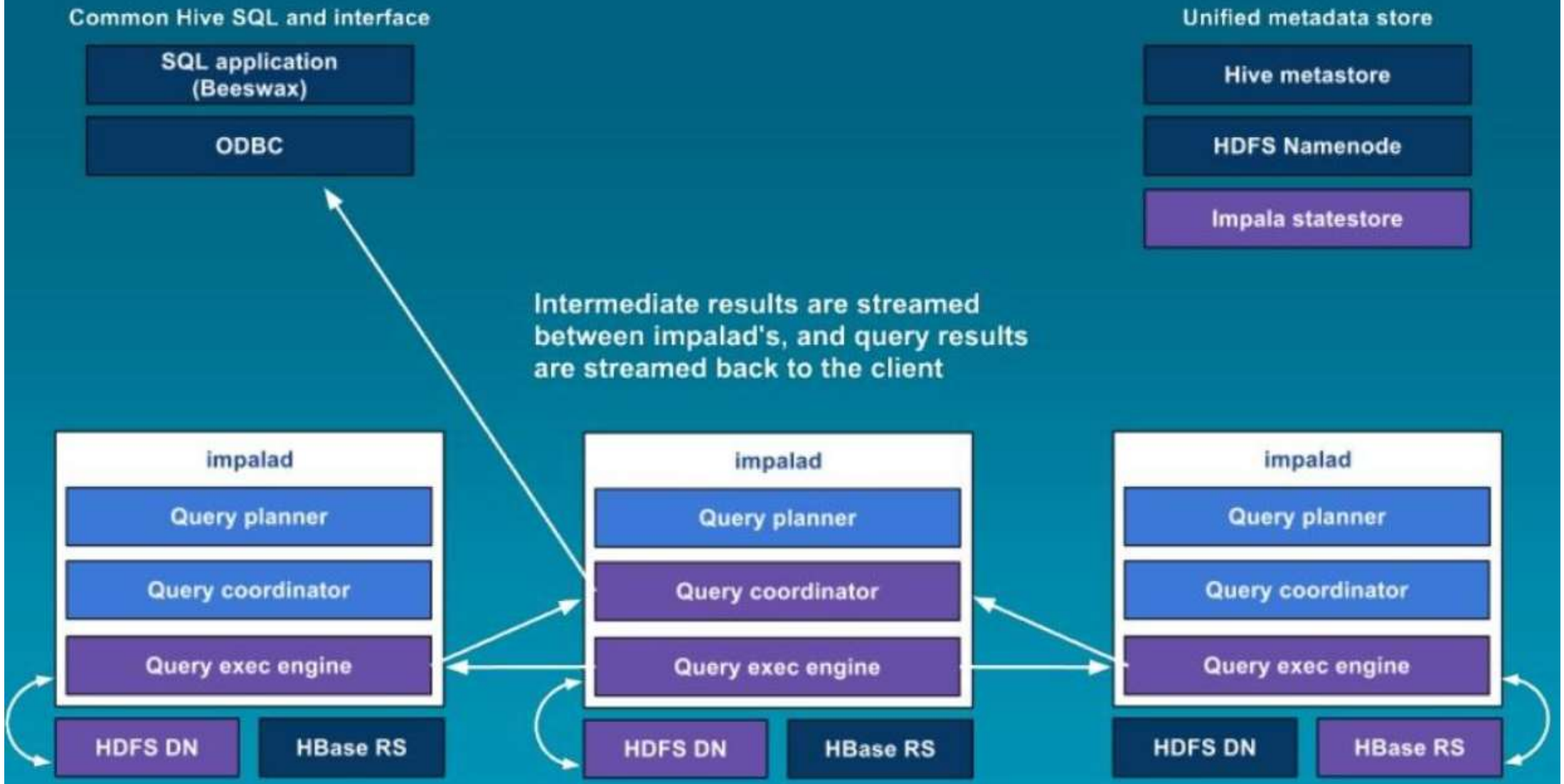
# Impala: Architecture



# Impala: Architecture



# Impala: Architecture



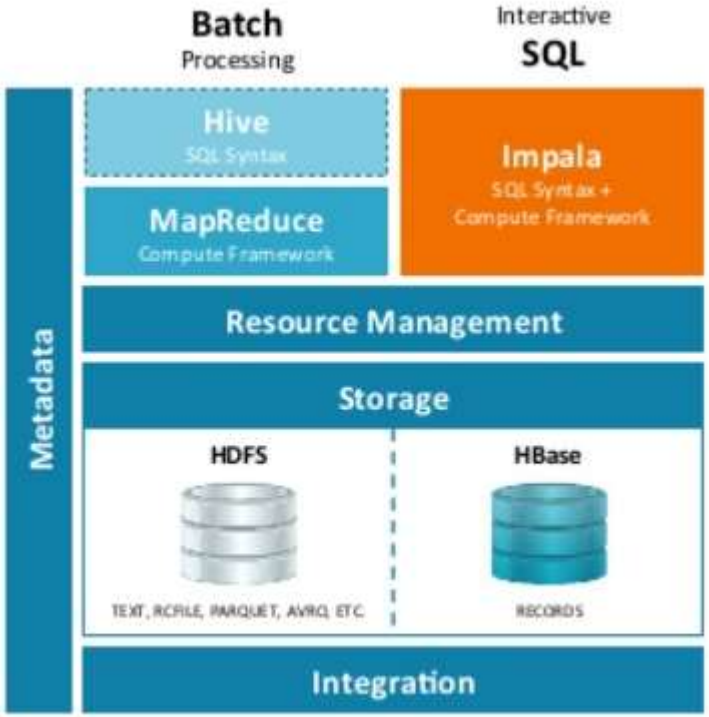
# Impala and Hive

## Shares everything Client-Facing

- Metadata ( Table definitions
- ODBC/JDBC Drives
- SQL Syntax ( Hive SQL)
- Flexible file Formats
- Machine pool
- Hue GUI

## But Built for Different Purposes

- Hive : runs on MapReduce and Ideal for Batch Processing
- Impala : native MPP query ideal for interactive SQL



# Comparing Hive and Impala

- Hive
  - MapReduce as an execution Engine
  - High latency, low throughput quarries
  - Fault-tolerance model based on MapReduce's on-disk check pointing, Materializes all intermediates results
  - Java runtime allows for easy late-binding of functionality for file format and UDFs
  - Extensive laying impose high overhead
- Impala
  - Direct, process-to-process data exchange
  - No fault tolerance
  - An execution engine designed for low runtime overhead

References

http://www.slideshare.net/dataera/how-impala-works-38586729

http://www.slideshare.net/arinto/apache-flume?qid=cccaf102-3256-416d-92af-59a46ca7fc22&v=&b=&from_search=5