Hive and Querying

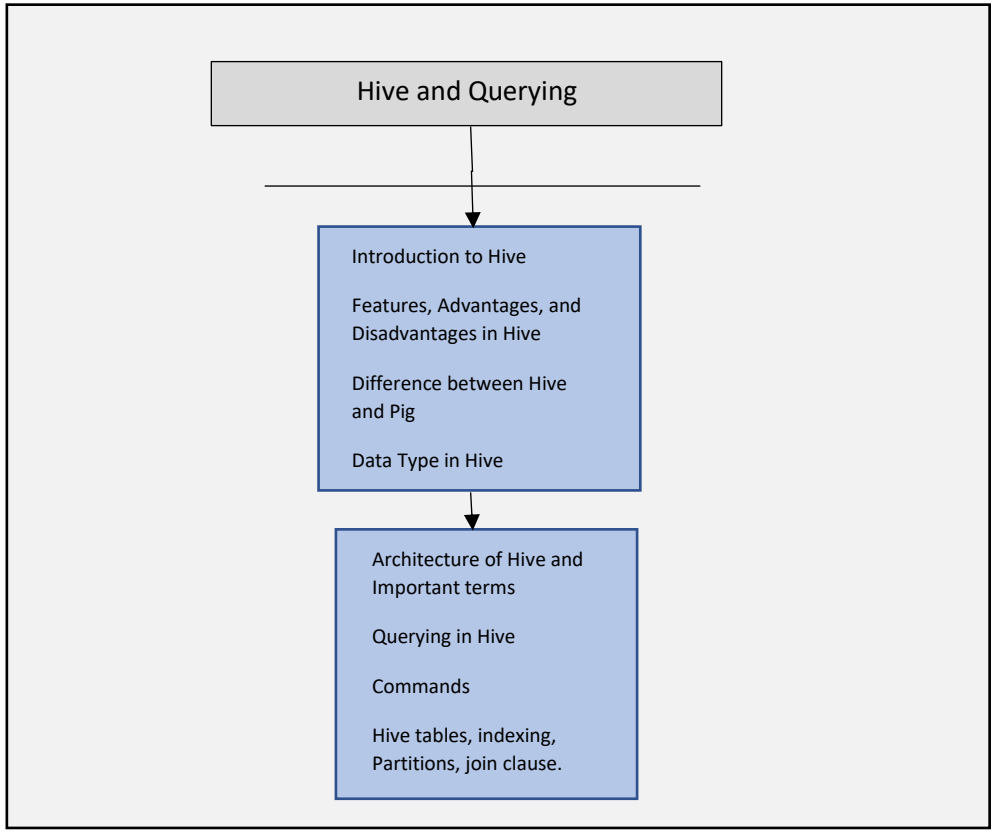
Hive is a data warehouse system which is used to analyse structured data. It runs SQL like queries called HQL (Hive query language) which gets internally converted to MapReduce jobs.

As a part of Hive and Querying, you covered:

- Introduction to Hive, Features, limitation, and advantages
- Difference between Hive and Pig, Hive Architecture, and Important terms
- Hive tables, Partition, Indexing and join clause in Hive
- Data type in Hive and Querying of Data in Hive

Common Interview Questions:

1. What applications are supported by Hive?
2. What are the different tables available in Hive?
3. What is the difference between external and managed tables?
4. Where does the data of a Hive table get stored?
5. Can Hive be used in OLTP systems?
6. Can a table name be changed in Hive?
7. Where is Hive table data stored?
8. Can we use the LOAD or INSERT command to view?
9. What is a Hive Metastore?



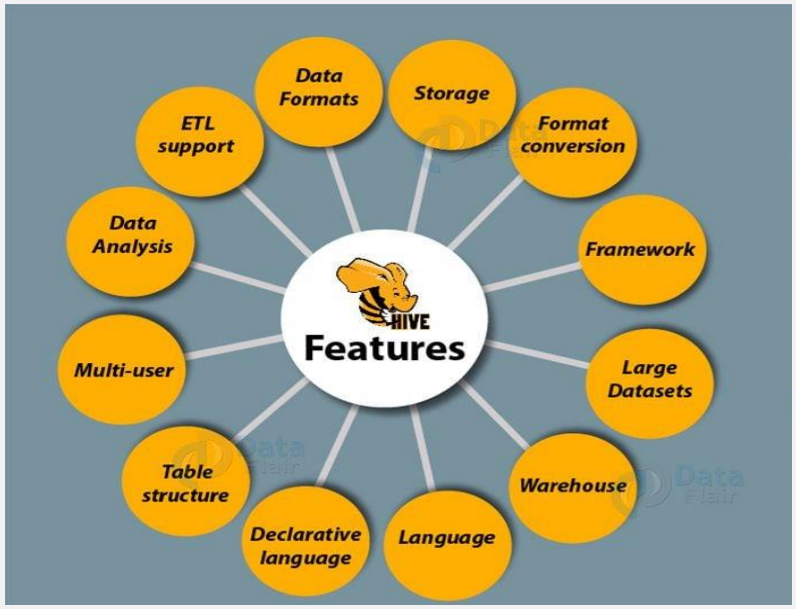
Hive and Querying

Apache Hive:

Hive is a data warehouse system which is used to analyse structured data. It is built on the top of Hadoop. It was developed by Facebook.

Hive provides the functionality of reading, writing, and managing large datasets residing in distributed storage. It runs SQL like queries called HQL (Hive query language) which gets internally converted to MapReduce jobs.

Features of Hive:

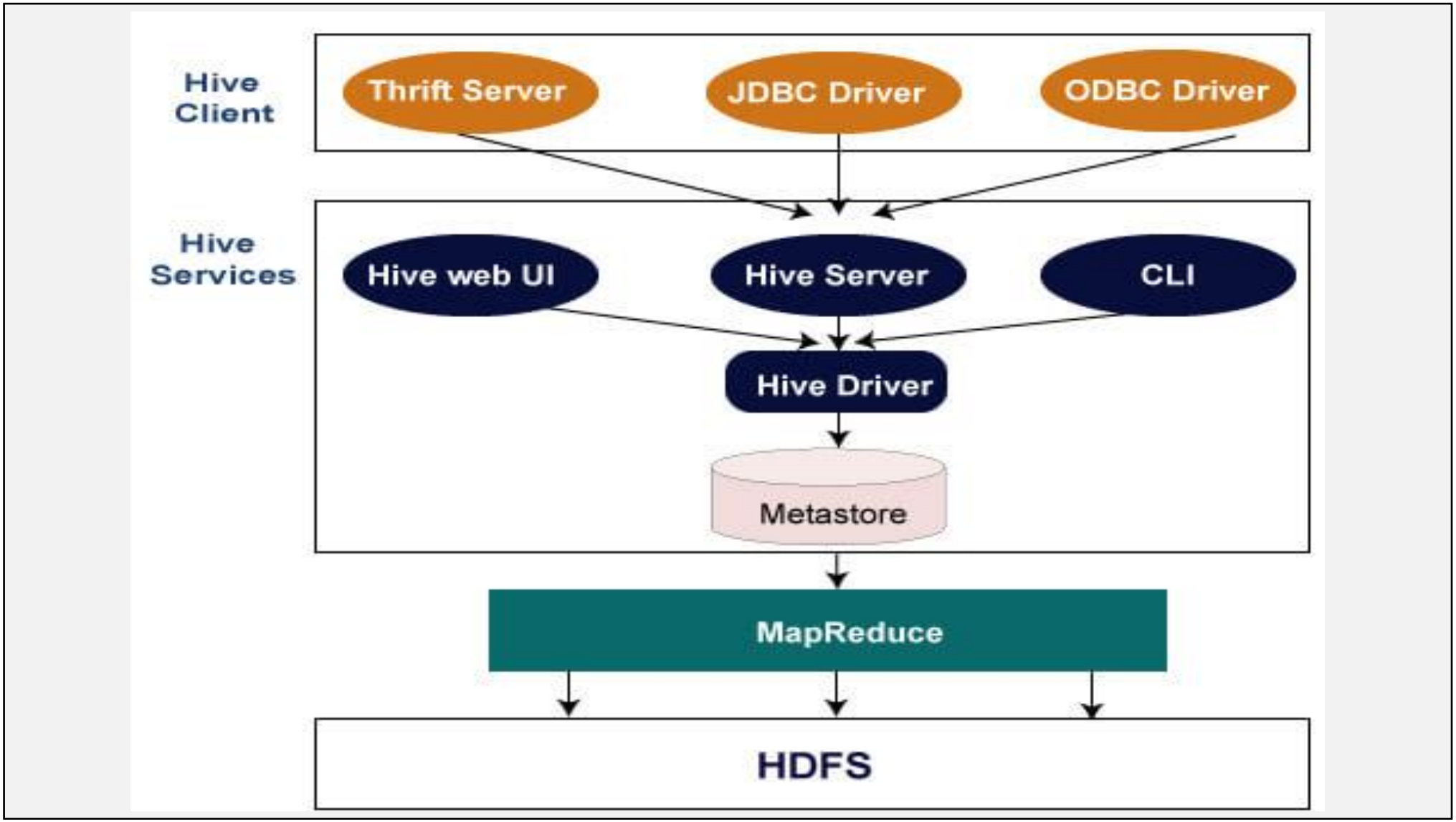


Apache Hive VS Pig:

Based on	Hive	Pig
Language	Hive uses a declarative language called HiveQL	With Pig Latin, a procedural data flow language is used.
Type of data	Hive works on structured data. Does not work on other types of data	Pig works on structured, semi-structured and unstructured data
Operates on	Works on the server-side of the cluster	Works on the client-side of the cluster
Avro File Format	Hive does not support Avro	Pig supports Avro
Schema	Hive supports schema	Creating schema is not required to store data in Pig
Data Processing	Hive is used for batch processing	Pig is a high-level data-flow language
JDBC/ ODBC	Supported, but limited	Unsupported
Loading Speed	Hive takes time to load but executes quickly	Pig loads data quickly

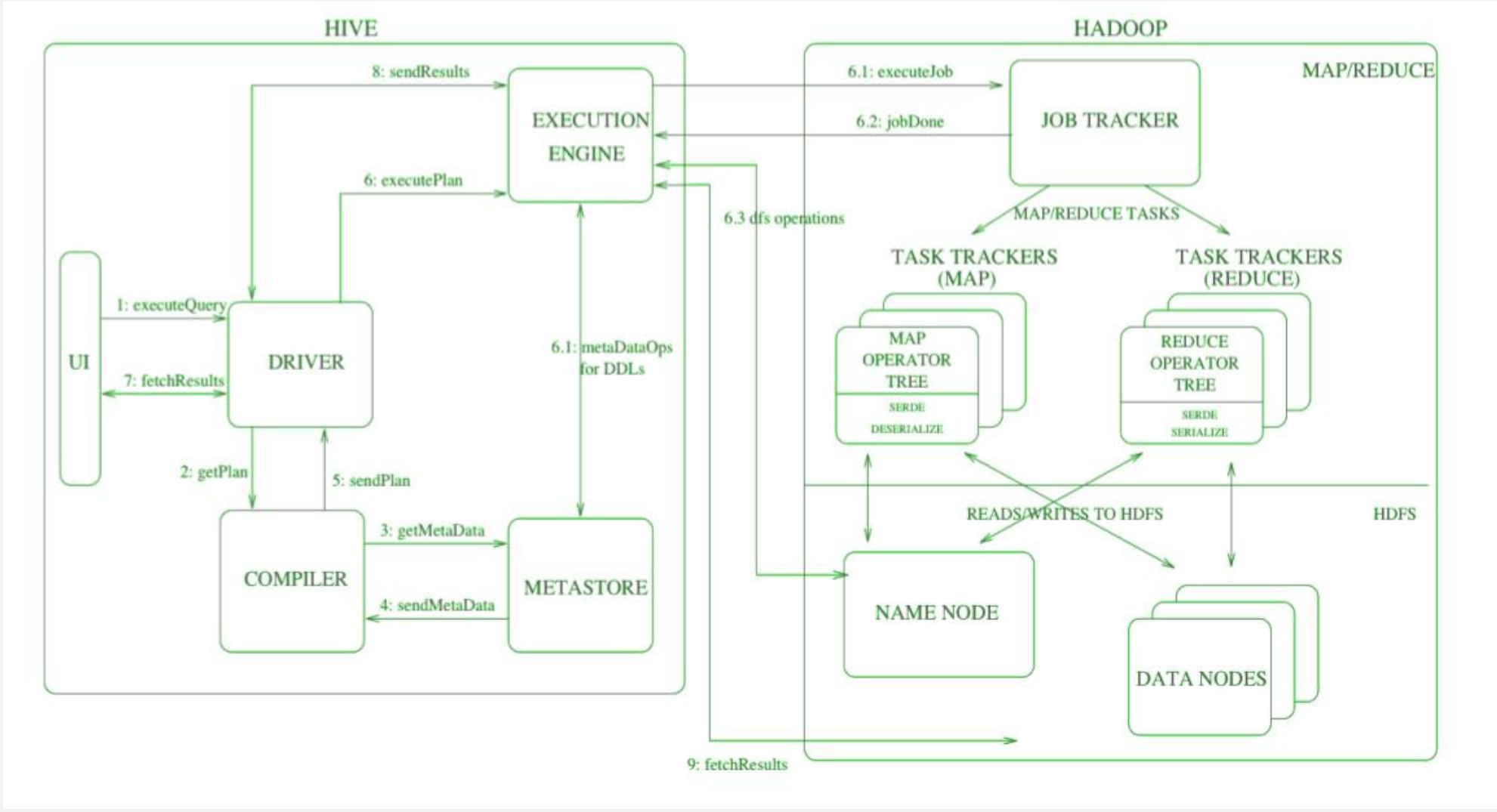
Hive and Querying

Hive Architecture:



Hive and Querying

Architecture of Hive that is built on the top of Hadoop:



Hive and Querying

Important Terminologies in Hive Architecture:

Hive Client:

Thrift Server - It is a cross-language service provider platform that serves the request from all those programming languages that supports Thrift.

JDBC Driver - It is used to establish a connection between hive and Java applications. The JDBC Driver is present in the class org.apache.hadoop.hive.jdbc.HiveDriver.

ODBC Driver - It allows the applications that support the ODBC protocol to connect to Hive.

Hive Services:

Hive CLI - The Hive CLI is a shell where we can execute Hive queries and commands.

Hive Web User Interface - The Hive Web UI is just an alternative of Hive CLI. It provides a web-based GUI for executing Hive queries and commands.

Hive MetaStore - It is a central repository that stores all the structure information of various tables and partitions in the warehouse. It also includes metadata of column and its type information, the serializers and deserializers which is used to read and write data and the corresponding HDFS files where the data is stored.

Hive Server - It is referred to as Apache Thrift Server. It accepts the request from different clients and provides it to Hive Driver.

Hive Driver - It receives queries from different sources like web UI, CLI, Thrift, and JDBC/ODBC driver. It transfers the queries to the compiler.

Hive Compiler - The purpose of the compiler is to parse the query and perform semantic analysis on the different query blocks and expressions. It converts HiveQL statements into MapReduce jobs.

Hive Execution Engine - Optimizer generates the logical plan in the form of DAG of map-reduce tasks and HDFS tasks. In the end, the execution engine executes the incoming tasks in the order of their dependencies.

Advantages and Disadvantages of using Hive:

Advantages of using Hive:

- Hive is built on Hadoop, which supports and handles all the capabilities of Hadoop provides like reliable, highly available, node failure, commodity hardware
- Database developers need not learn Java programming for writing map-reduce programs for retrieving data from the Hadoop system.
- Data stored in HDFS so you will have features of scalability, redundancy over hive SQL language
- Querying data using hive is simple and easy to use

Disadvantages of using Hive:

- Hive is not for OLAP processing, only supports OLTP processing
- Subqueries are not supported.
- It has a high latency.
- Hive tables don't support delete or update operations.
- ACID transaction limitation on Hive
- indexing limitations,
- schema operation

Hive Data Type:

Integer Types:

Type	Size	Range
TINTINT	1-byte signed integer	-128 to 127
SMALLINT	2-byte signed integer	32,768 to 32,767
INT	4-byte signed integer	2,147,483,648 to 2,147,483,647
BIGINT	8-byte signed integer	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807

String Types:

STRING

The string is a sequence of characters. It values can be enclosed within single quotes (') or double quotes (").

Varchar

The varchar is a variable length type whose range lies between 1 and 65535.

CHAR

The char is a fixed-length type whose maximum length is fixed at 255.

Date/Time Types:

TIMESTAMP: Timestamp format "YYYY-MM-DD HH:MM:SS.fffffffff" (9 decimal place precision)

DATES: The Date value is used to specify a particular year, month and day, in the form YYYY--MM--DD.

Complex Types:

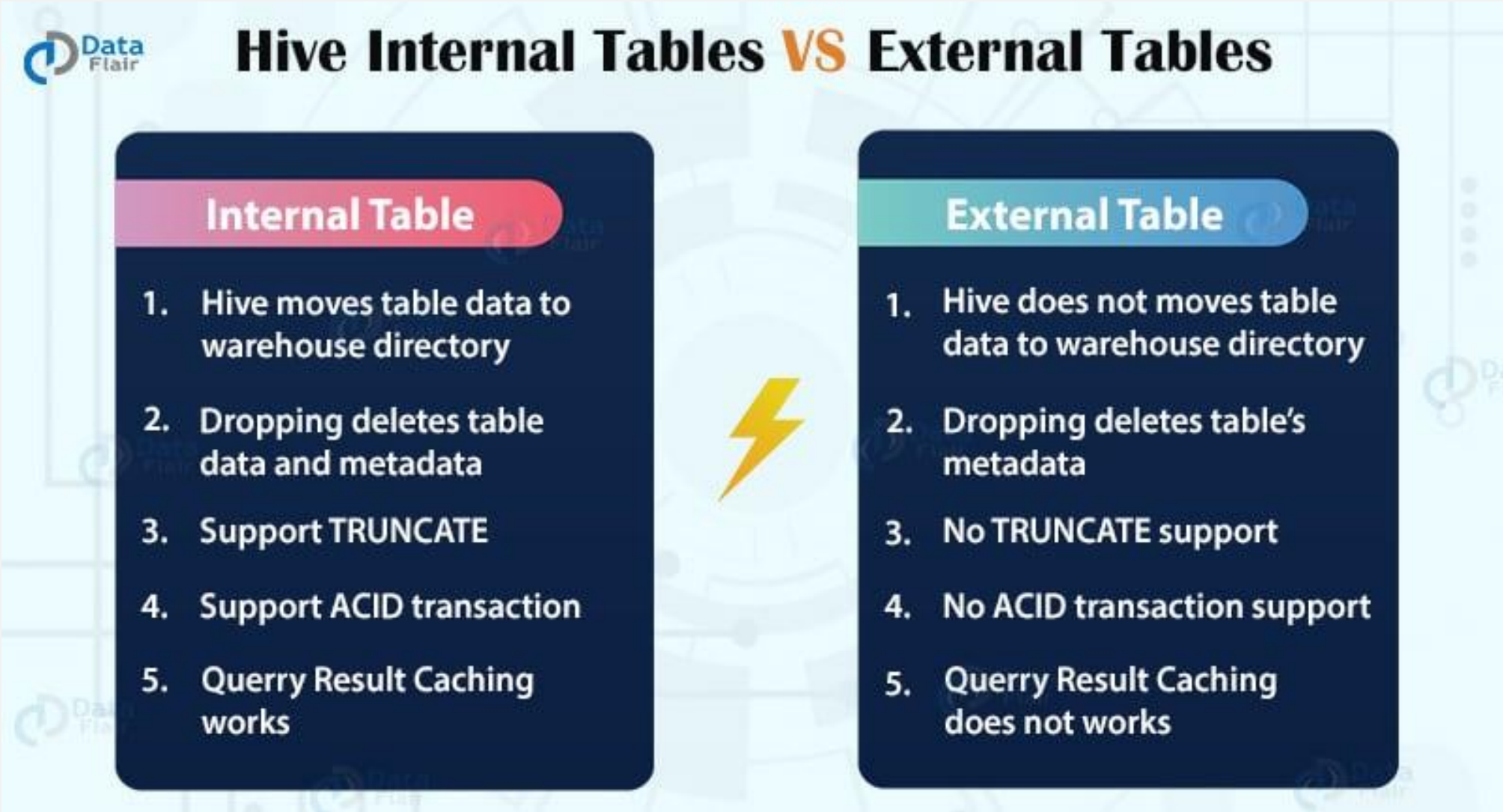
Type	Range
Struct	struct('James','Roy')
Map	map('first','James','last','Roy')
Array	array('James','Roy')

Decimal Types:

Type	Range
Float	Single precision floating point number
Double	Double precision floating point number

Hive and Querying

Types of Hive Table:



Hive and Querying

Data Querying with Hive:

Data Querying with Hive:

Hive provides SQL type querying language for the ETL purpose on top of Hadoop file system. Hive Query language (HiveQL) provides SQL type environment in Hive to work with tables, databases, queries. We can have a different type of Clauses associated with Hive to perform different type data manipulations and querying.

Hive DML Commands:

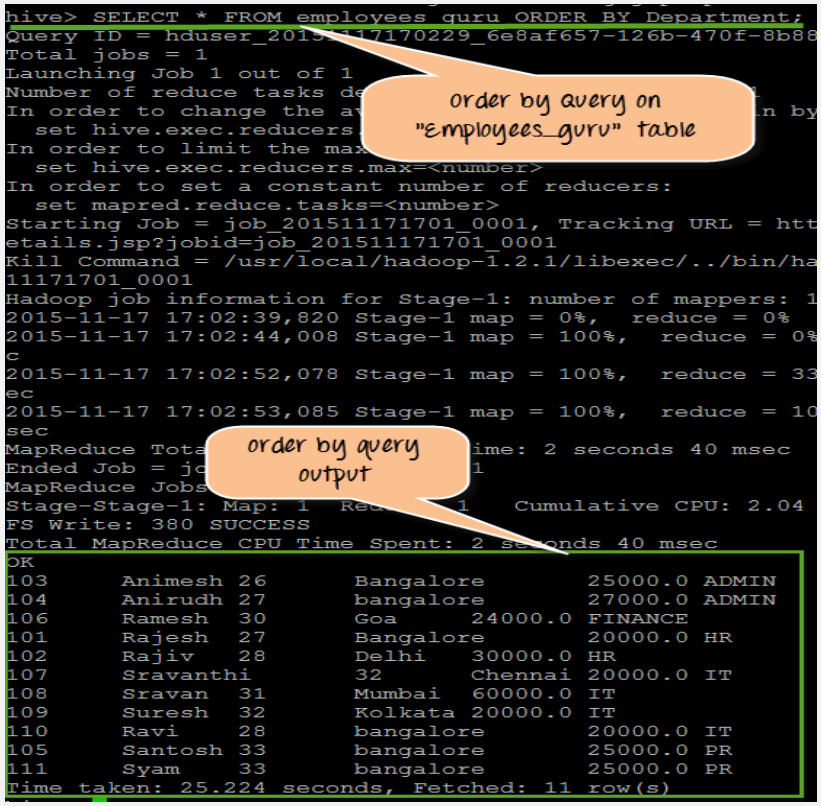
Hive DDL Commands:

Hive and Querying

Order By:

The ORDER BY syntax in HiveQL uses the "SELECT" statement to help sort data. The query will only pick the column name mentioned in the Order by clause, and display the matching column values in ascending or descending order.

Example:

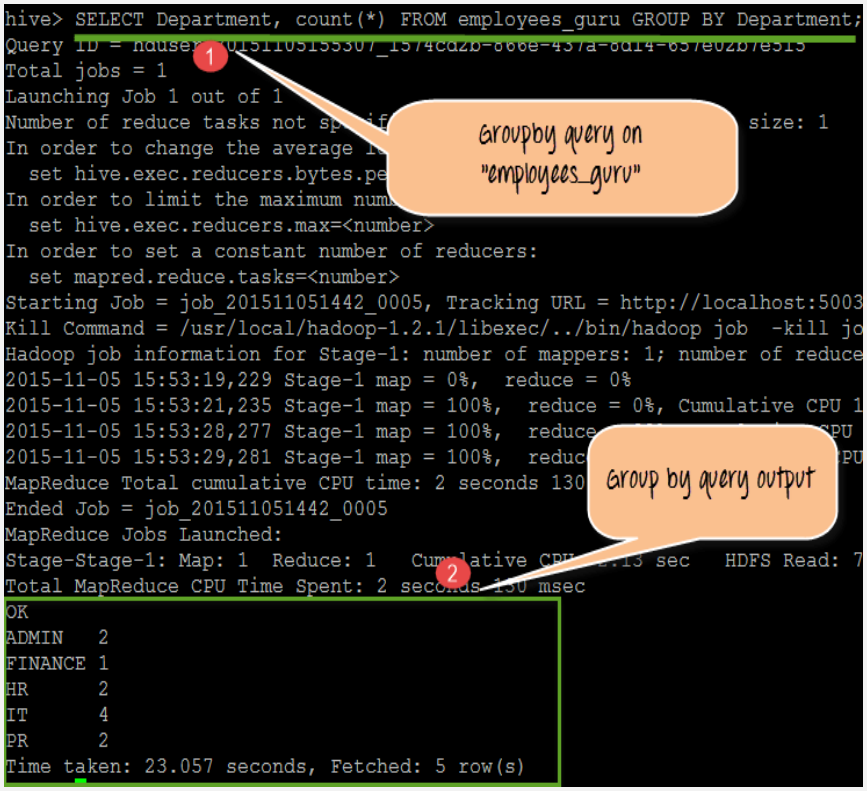



```
hive> SELECT * FROM employees_guru ORDER BY Department;
Query ID = hduser_20151117170229_6e8af657-126b-470f-8b88
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks de
In order to change the av                                  n by
  set hive.exec.reducers
In order to limit the max
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapred.reduce.tasks=<number>
Starting Job = job_201511171701_0001, Tracking URL = htt
etails.jsp?jobid=job_201511171701_0001
Kill Command = /usr/local/hadoop-1.2.1/libexec/../bin/ha
11171701_0001
Hadoop job information for Stage-1: number of mappers: 1
2015-11-17 17:02:39,820 Stage-1 map = 0%,  reduce = 0%
2015-11-17 17:02:44,008 Stage-1 map = 100%,  reduce = 0%
c
2015-11-17 17:02:52,078 Stage-1 map = 100%,  reduce = 33
ec
2015-11-17 17:02:53,085 Stage-1 map = 100%,  reduce = 10
sec
MapReduce Tota                                ime: 2 seconds 40 msec
Ended Job = jo                                1
MapReduce Jobs
Stage-Stage-1: Map: 1  Re                     1   Cumulative CPU: 2.04
FS Write: 380 SUCCESS
Total MapReduce CPU Time Spent: 2 seconds 40 msec
OK
103     Animesh 26      Bangalore       25000.0 ADMIN
104     Anirudh 27      bangalore       27000.0 ADMIN
106     Ramesh  30      Goa     24000.0 FINANCE
101     Rajesh  27      Bangalore       20000.0 HR
102     Rajiv   28      Delhi   30000.0 HR
107     Sravanthi       32      Chennai 20000.0 IT
108     Sravan  31      Mumbai  60000.0 IT
109     Suresh  32      Kolkata 20000.0 IT
110     Ravi    28      bangalore       20000.0 IT
105     Santosh 33      bangalore       25000.0 PR
111     Syam    33      bangalore       25000.0 PR
Time taken: 25.224 seconds, Fetched: 11 row(s)
```

Group by:

The query will only look at the columns under the name defined as "group by" clause, and it will show the results by grouping the specific and matching column values.

Example:



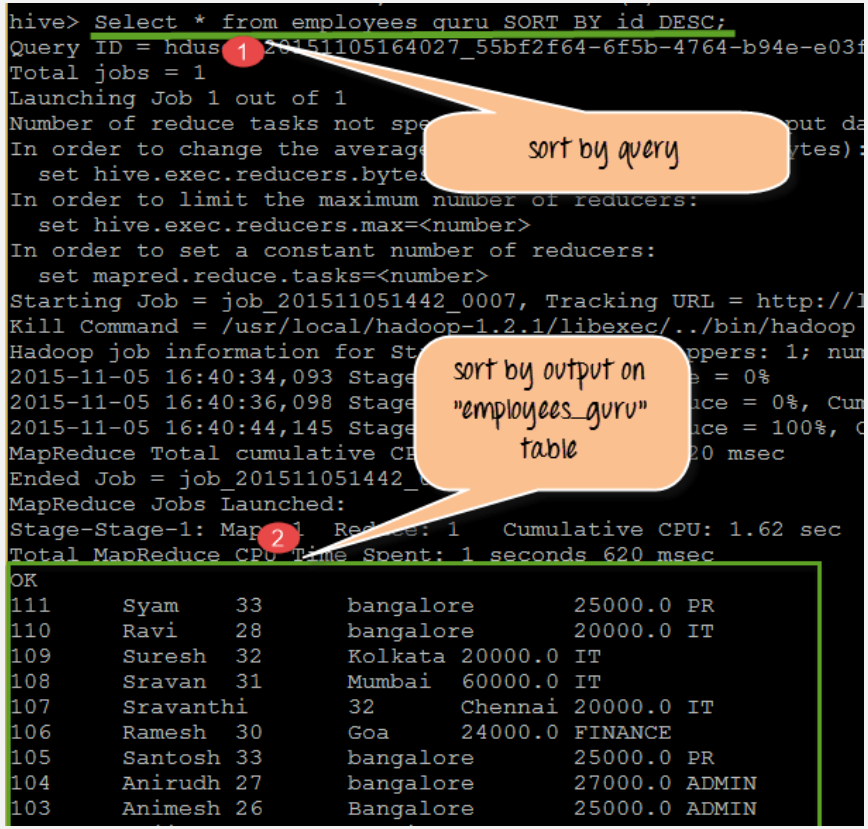

```
hive> SELECT Department, count(*) FROM employees_guru GROUP BY Department;
Query ID = hduser_20151105155307_1574cd2b-866e-437a-8d14-657e0287e515
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks not sp                              size: 1
In order to change the average l
  set hive.exec.reducers.bytes.pe
In order to limit the maximum num
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapred.reduce.tasks=<number>
Starting Job = job_201511051442_0005, Tracking URL = http://localhost:5003
Kill Command = /usr/local/hadoop-1.2.1/libexec/../bin/hadoop job  -kill jo
Hadoop job information for Stage-1: number of mappers: 1; number of reduce
2015-11-05 15:53:19,229 Stage-1 map = 0%,  reduce = 0%
2015-11-05 15:53:21,235 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 1
2015-11-05 15:53:28,277 Stage-1 map = 100%,  reduce                    CPU
2015-11-05 15:53:29,281 Stage-1 map = 100%,  reduc                     CPU
MapReduce Total cumulative CPU time: 2 seconds 130
Ended Job = job_201511051442_0005
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU                .13 sec   HDFS Read: 7
Total MapReduce CPU Time Spent: 2 seconds 130 msec
OK
ADMIN   2
FINANCE 1
HR      2
IT      4
PR      2
Time taken: 23.057 seconds, Fetched: 5 row(s)
```

Hive and Querying

Sort By:

When a Hive query comes with a “Sort by” clause, it goes through the columns under the name defined by the query. Once executed, the query explores columns of Hive tables to sort the output.

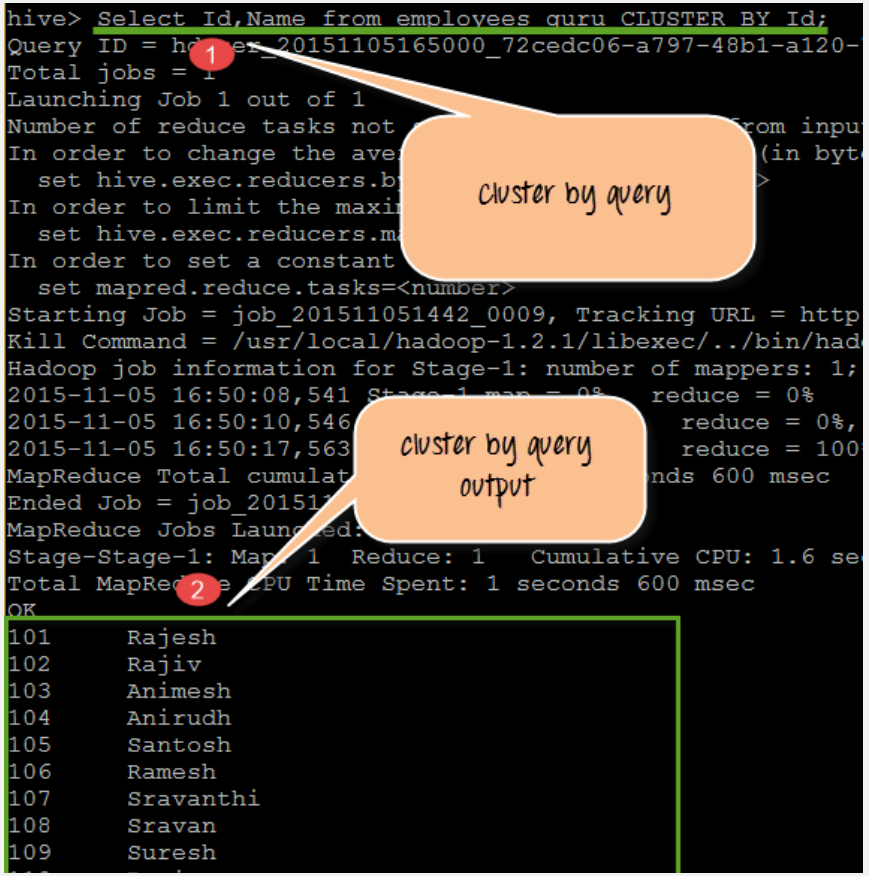
Example:



Cluster By:

Hive queries with a CLUSTER BY clause or command are typically deployed in queries to perform the functions of both DISTRIBUTE BY and SORT BY together.

Example:



Hive and Querying

Distribute By:

The DISTRIBUTE BY instruction determines how the output is divided among reducers in a MapReduce job.

Example:

Some effective ways to optimize Hive queries:

Partition keys present an opportunity to target a subset of the table data, rather than scan data you don't need for your operations.

Bucketing allows you to target a subset of data. It improves join performance specifically by scanning fewer data.

Compression minimizes the amount of data that traverses each of those steps and decreases the time spent moving through the query states.

Eliminating small file operations from your query promotes a healthy Hive ecosystem. Each file is tracked by the Hive metastore and stored in HDFS, which are each performance-optimized to handle larger files over many smaller files.

Hive and Querying

Indexes in Hive:

Indexes in Hive:

Indexes are a pointer or reference to a record in a table as in relational databases. In Hive, the index table is different than the main table. Indexes facilitate in making query execution or search operation faster.

There are two types of indexes in Hive:

Bitmap Indexing: This is used with columns having a few distinct values. It is known to store both the indexed column's value and the list of rows as a bitmap.

Compact Indexing: This type of indexing is known to store the column value and storage blockid.

Hive Partitions:

Hive Partitions:

Apache Hive organizes tables into partitions. Partitioning is a way of dividing a table into related parts based on the values of particular columns like date, city, and department.

There are two types of Partitioning in Hive:

Static Partitioning: Insert input data files individually into a partition table is Static Partition.

Dynamic Partitioning: Single insert to partition table is known as a dynamic partition.

JOIN Clause in Hive:

JOIN Clause in Hive:

JOIN is a clause that is used for combining specific fields from two tables by using values common to each one. It is used to combine records from two or more tables in the database.

SYNTAX:

```
join_table:

    table_reference JOIN table_factor [join_condition]
    | table_reference {LEFT|RIGHT|FULL} [OUTER] JOIN table_reference
    join_condition
    | table_reference LEFT SEMI JOIN table_reference join_condition
    | table_reference CROSS JOIN table_reference [join_condition]
```