Chapter 9

Introduction to Hive

Learning Objectives and Learning Outcomes

Learning Objectives	Learning Outcomes
Introduction to Hive 1. To study the Hive Architecture 2. To study the Hive File format 3. To study the Hive Query Language	a) To understand the hive architecture. b) To create databases, tables and execute data manipulation language statements on it. c) To differentiate between static and dynamic partitions. d) To differentiate between managed and external tables.

Agenda

- What is Hive?
- Hive Architecture
- Hive Data Types
- Primitive Data Types
- Collection Data Types
- Hive File Format
  - Text File
  - Sequential File
  - RCFile (Record Columnar File)

Agenda

- Hive Query Language
  - DDL (Data Definition Language) Statements
  - DML (Data Manipulation Language) Statements
  - Database
  - Tables
  - Partitions
  - Buckets
  - Aggregation
  - Group BY and Having
- SERDER

What is Hive?

What is Hive?

- Data Warehousing tool (Data Warehouses are central repositories of integrated **data** from one or more disparate sources)
- Facilitates querying and managing large datasets residing in distributed storage
- Facebook created Hive component to manage their ever-growing volumes of data.
- Hive makes use of the following:
    1. HDFS for Storage
    2. MapReduce for execution
    3. Stores metadata in an RDBMS.

Hive Features

Features of Hive

1. It is similar to SQL.
2. HQL is easy to code.
3. Hive supports rich data types such as structs, lists, and maps.
4. Hive supports SQL filters, group-by and order-by clauses.

Hive Data Types

Hive Data Units

- **Databases** – Namespaces that separate tables and other data units from naming confliction

- **Tables** – Set of records that have similar schema

- **Partitions** – Logical separations of data based on classification of given information as per specific attributes.

    - For example, a date_partition of type STRING and country_partition of type STRING. Each unique value of the partition keys defines a partition of the Table. For example all "US" data from "2009-12-23" is a partition of the page_views table. Therefore, if you run analysis on only the "US" data for 2009-12-23, you can run that query only on the relevant partition of the table thereby speeding up the analysis significantly.

- **Buckets (Clusters)** Data in each partition may in turn be divided into Buckets based on the value of a hash function of some column of the Table.
    - For example the page_views table may be bucketed by userid, which is one of the columns

Let say sample table is partitioned by 'State' column and clustered by 'Age' column of 2 buckets

In warehouse, the data is stored as

/user/$USER/warehouse/State=AL/part-00000

/user/$USER/warehouse/State=AL/part-00001

/user/$USER/warehouse/State=GA/part-00000

/user/$USER/warehouse/State=GA/part-00001

.

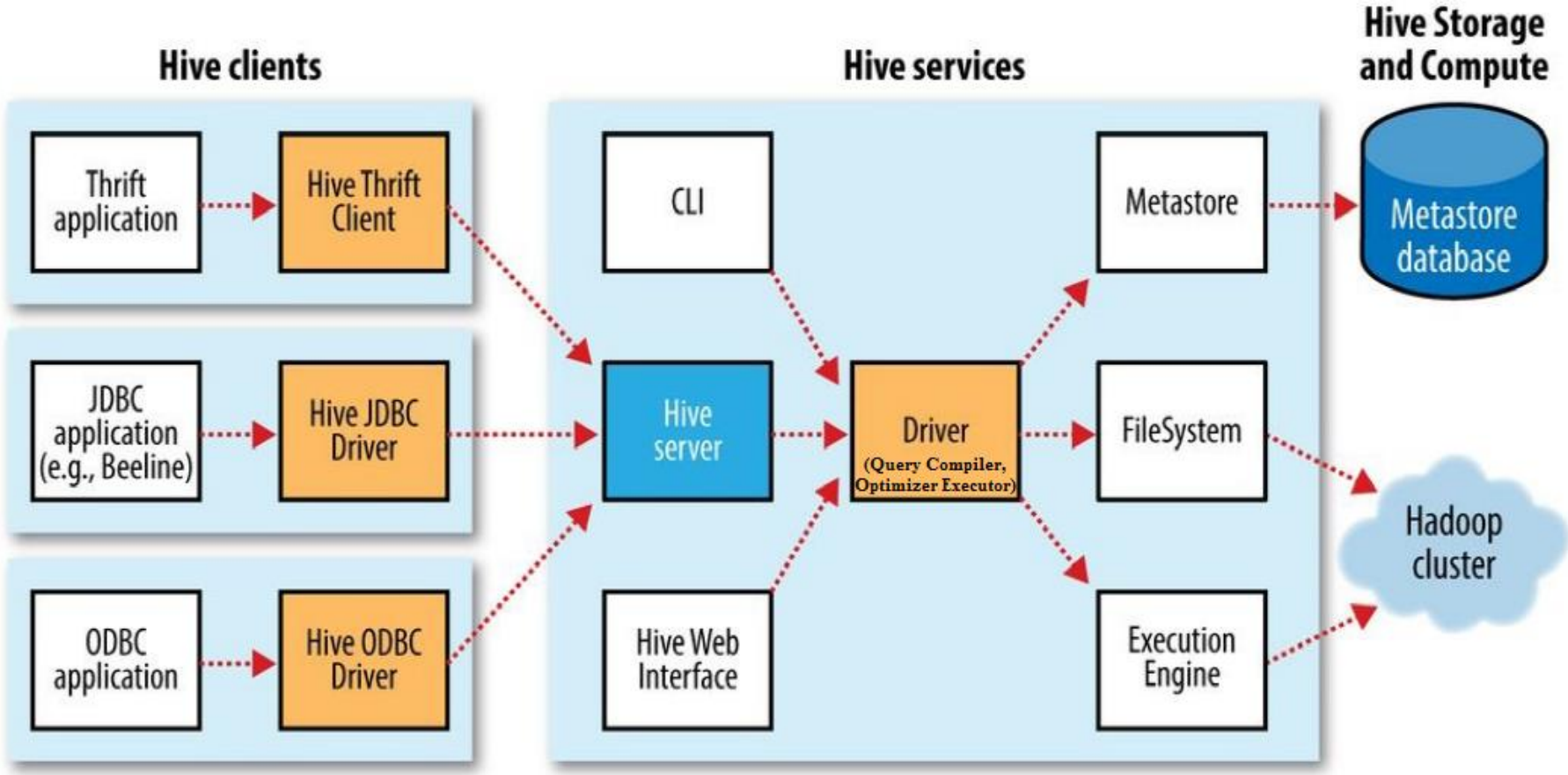
.

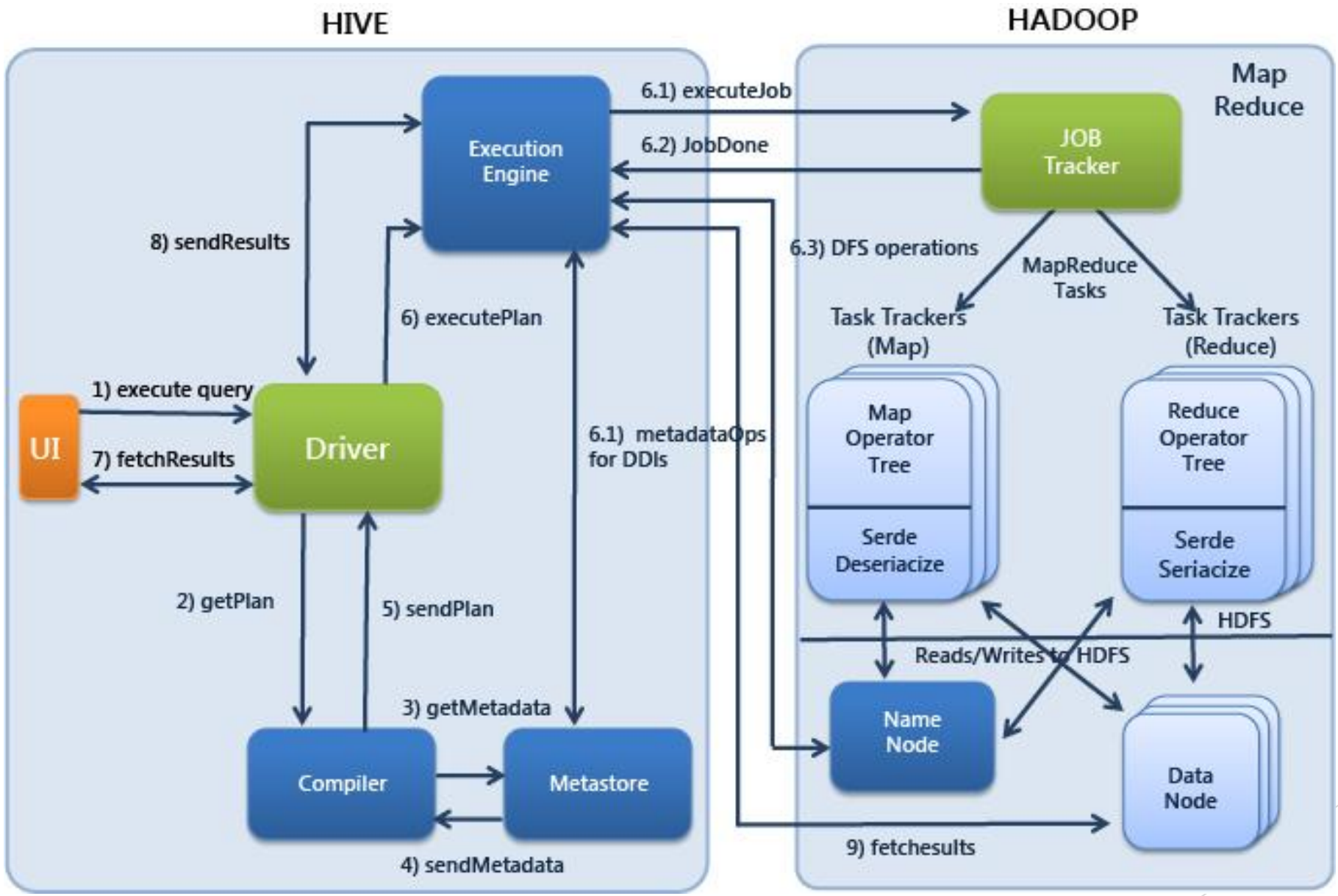
/user/$USER/warehouse/State=ND/part-00000

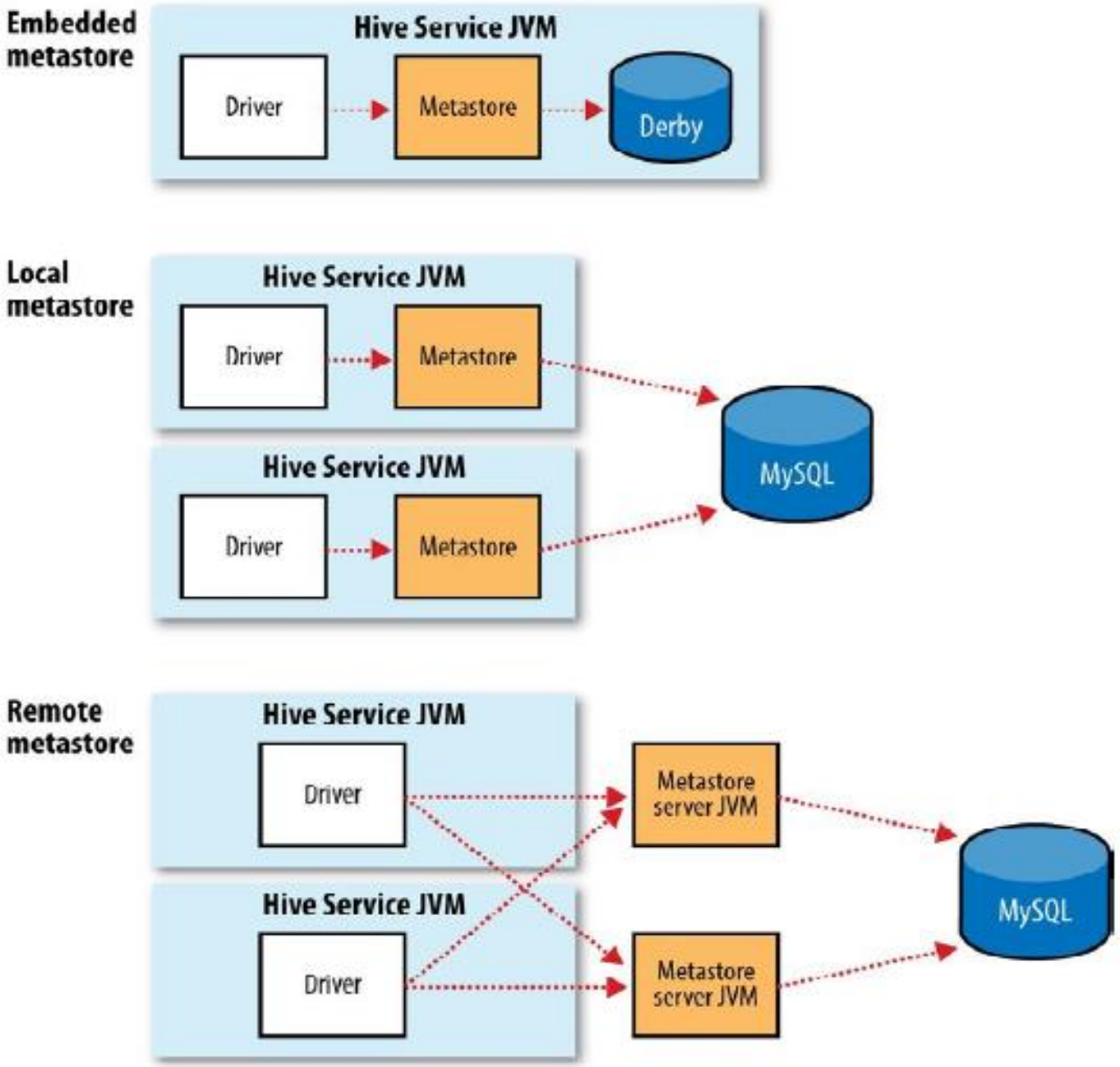
/user/$USER/warehouse/State=ND/part-00001

Hive Architecture

Hive Architecture







Hive Data Types

Hive Data Types

Numeric Data Type	
TINYINT	1 - byte signed integer
SMALLINT	2 -byte signed integer
INT	4 - byte signed integer
BIGINT	8 - byte signed integer
FLOAT	4 - byte single-precision floating-point
DOUBLE	8 - byte double-precision floating-point number

String Types	
STRING	
VARCHAR	Only available starting with Hive 0.12.0
CHAR	Only available starting with Hive 0.13.0
Strings can be expressed in either single quotes (‘) or double quotes (“)	

Miscellaneous Types	
BOOLEAN	
BINARY	Only available starting with Hive

Hive Data Types

Collection Data Types	
STRUCT	Similar to ‘C’ struct. Fields are accessed using dot notation. E.g.: struct('John', 'Doe')
MAP	A collection of key - value pairs. Fields are accessed using [] notation. E.g.: map('first', 'John', 'last', 'Doe')
ARRAY	Ordered sequence of same types. Fields are accessed using array index. E.g.: array('John', 'Doe')