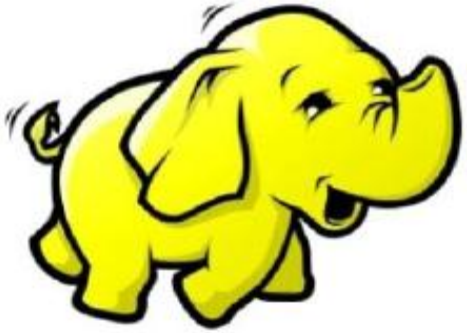
Big Data & Hadoop

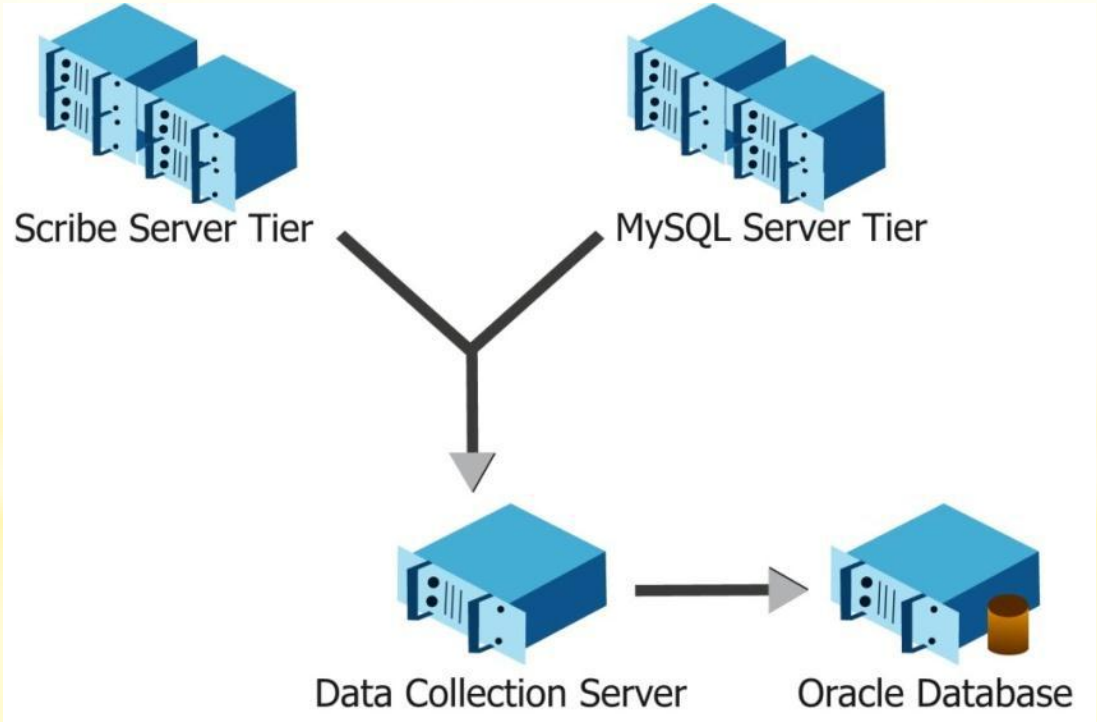


Topics

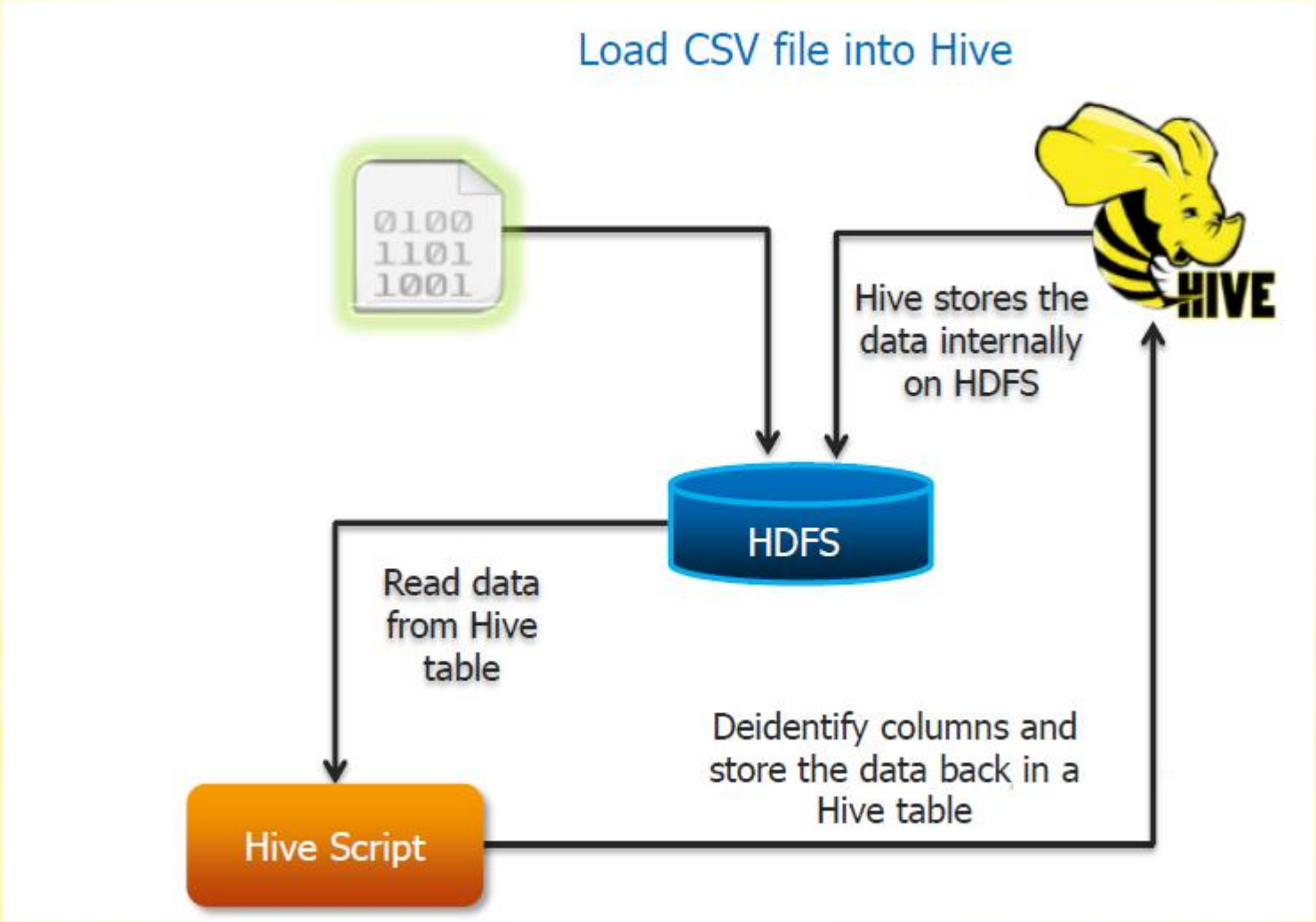
- ✓ What is Hive?
- ✓ Where to use Hive
- ✓ Why go for Hive when PIG is there?
- ✓ Let’s start -Hive Architecture
- ✓ Hive Components
- ✓ Hive Background
- ✓ How Facebook uses Hive
- ✓ Limitation of Hive
- ✓ Abilities of Hive Query Language
- ✓ Differences with Traditional RDBMS
- ✓ Hive Types
- ✓ Examples

Hive Background

- ✓ Started at **Facebook**.
- ✓ Data was collected by nightly cronjobs into **OracleDB**.
- ✓ **"ETL"** via hand-coded python.
- ✓ Grew from **10s of GBs** (2006) to **1 TB/day** new data (2007), now 10x that.



Use Case in Healthcare



Hive

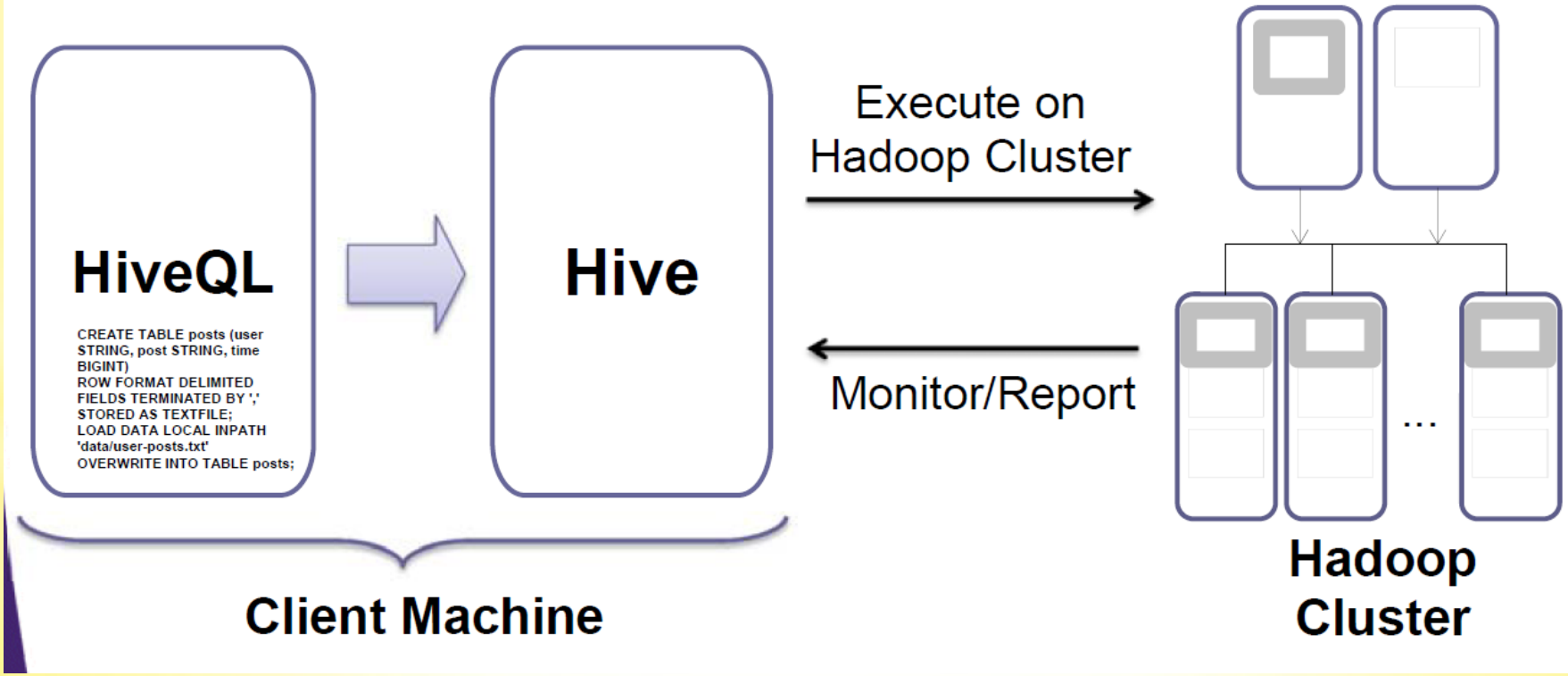
- Data Warehousing Solution built on top of Hadoop
- Provides SQL-like query language named HiveQL
  - Minimal learning curve for people with SQL expertise
  - Data analysts are target audience
- Early Hive development work started at Facebook in 2007
- Today Hive is an Apache project under Hadoop
- Ability to bring structure to various data formats
- Simple interface for ad hoc querying, analyzing and summarizing large amounts of data
- Access to files on various data stores such as HDFS and Hbase
- Hive does NOT provide low latency or real time queries
- Even querying small amounts of data may take minutes
- Designed for scalability and ease-of-use rather than low latency responses

What is Hive ?

- ✓ Data Warehousing package built on top of Hadoop
- ✓ It is similar to SQL and called HiveQL
- ✓ Used for data analysis
- ✓ For managing and querying structured data
- ✓ Targeted towards users comfortable with SQL
- ✓ Abstracts complexity of Hadoop
- ✓ No need learn java and Hadoop APIs
- ✓ Developed by Facebook and contributed to community
- ✓ Facebook analyzed several Terabytes of data everyday using Hive

HIVE

- **Translates HiveQL statements into a set of MapReduce Jobs which are then executed on a Hadoop Cluster**



What is HIVE ?

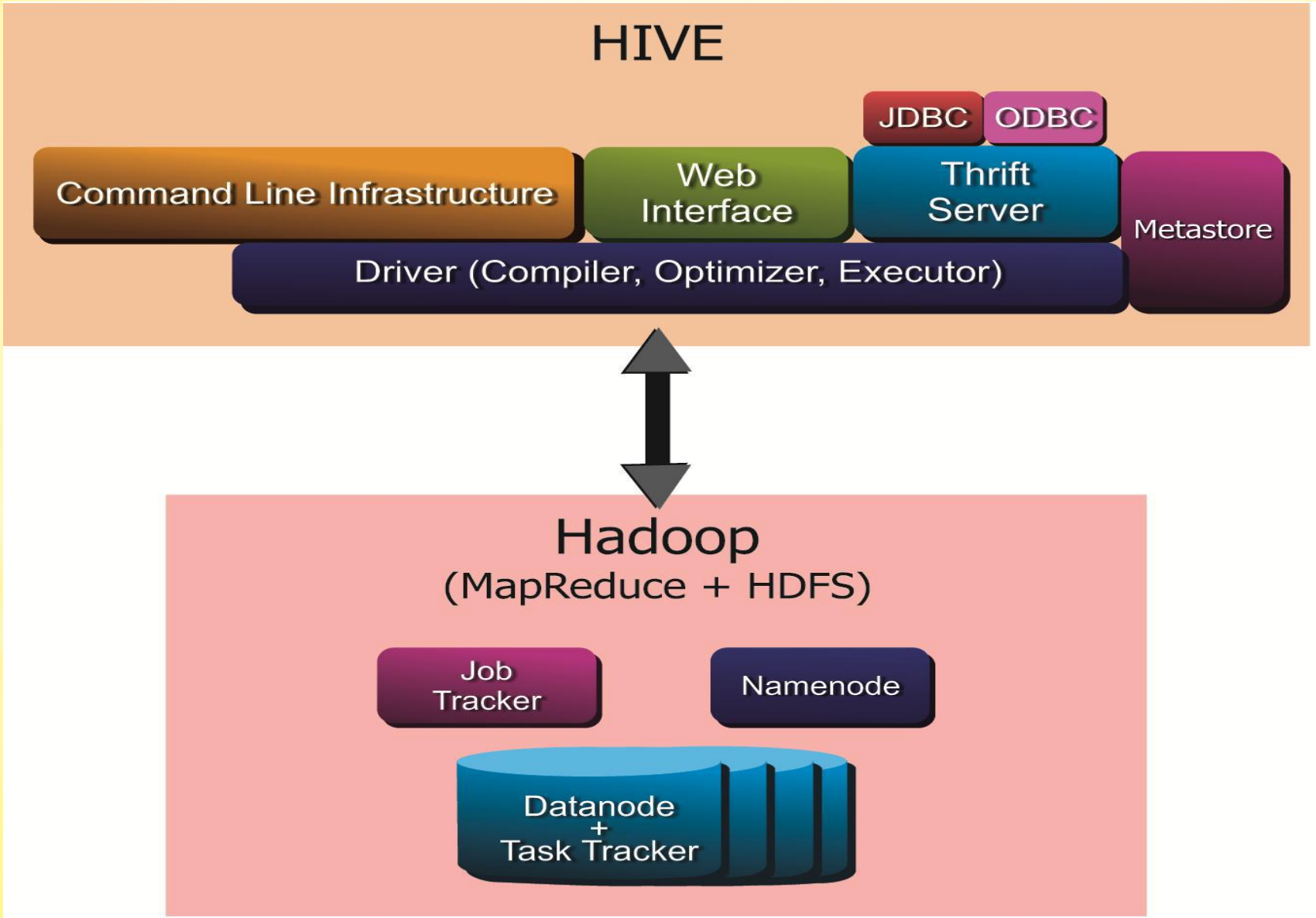
Defines SQL-Like query language called QL

Data Warehouse Infrastructure

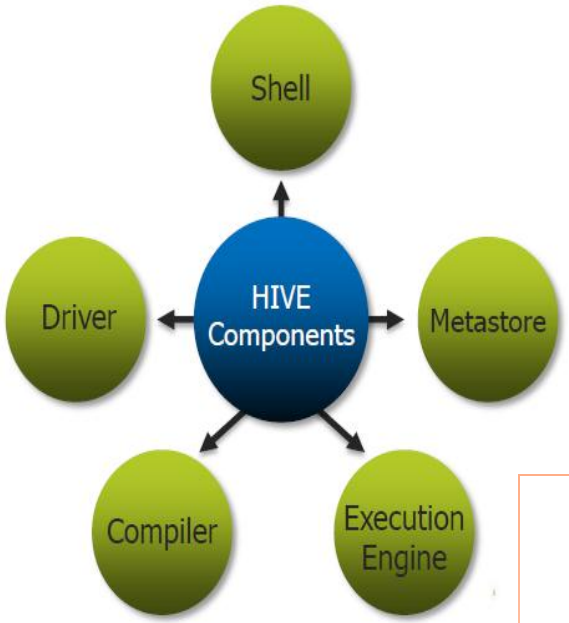
Allows programmers to plug-in custom mappers and reducers

It provides tools to enable easy data ETL

Hive Architecture



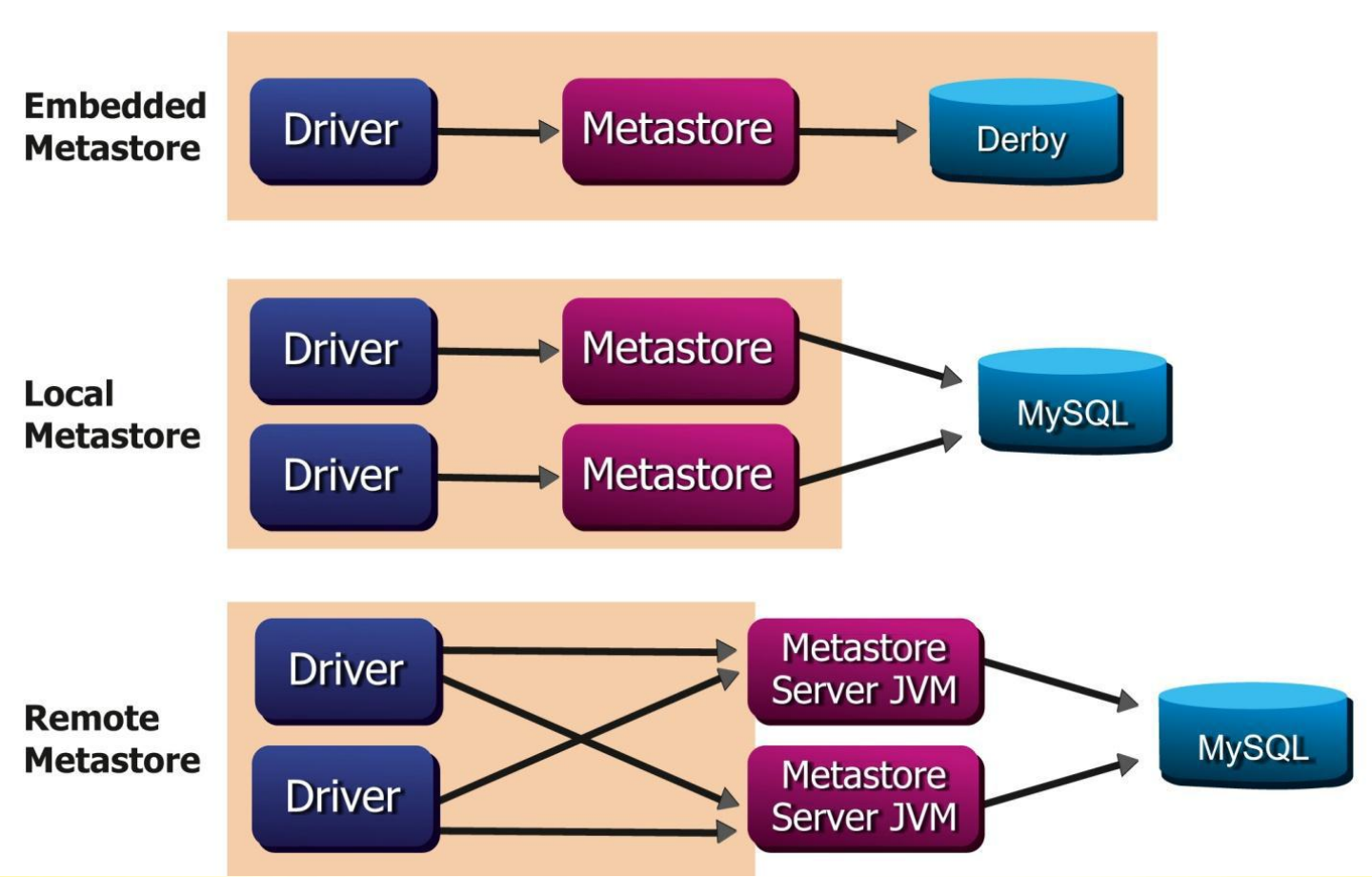
Hive Components



- Shell: allows interactive queries like MySQL shell connected to database
  - Also supports web and JDBC clients
- Driver: session handles, fetch, execute
- Compiler: parse, plan, optimize
- Execution engine: DAG of stages (M/R, HDFS, or metadata)
- Metastore: schema, location in HDFS, SerDe

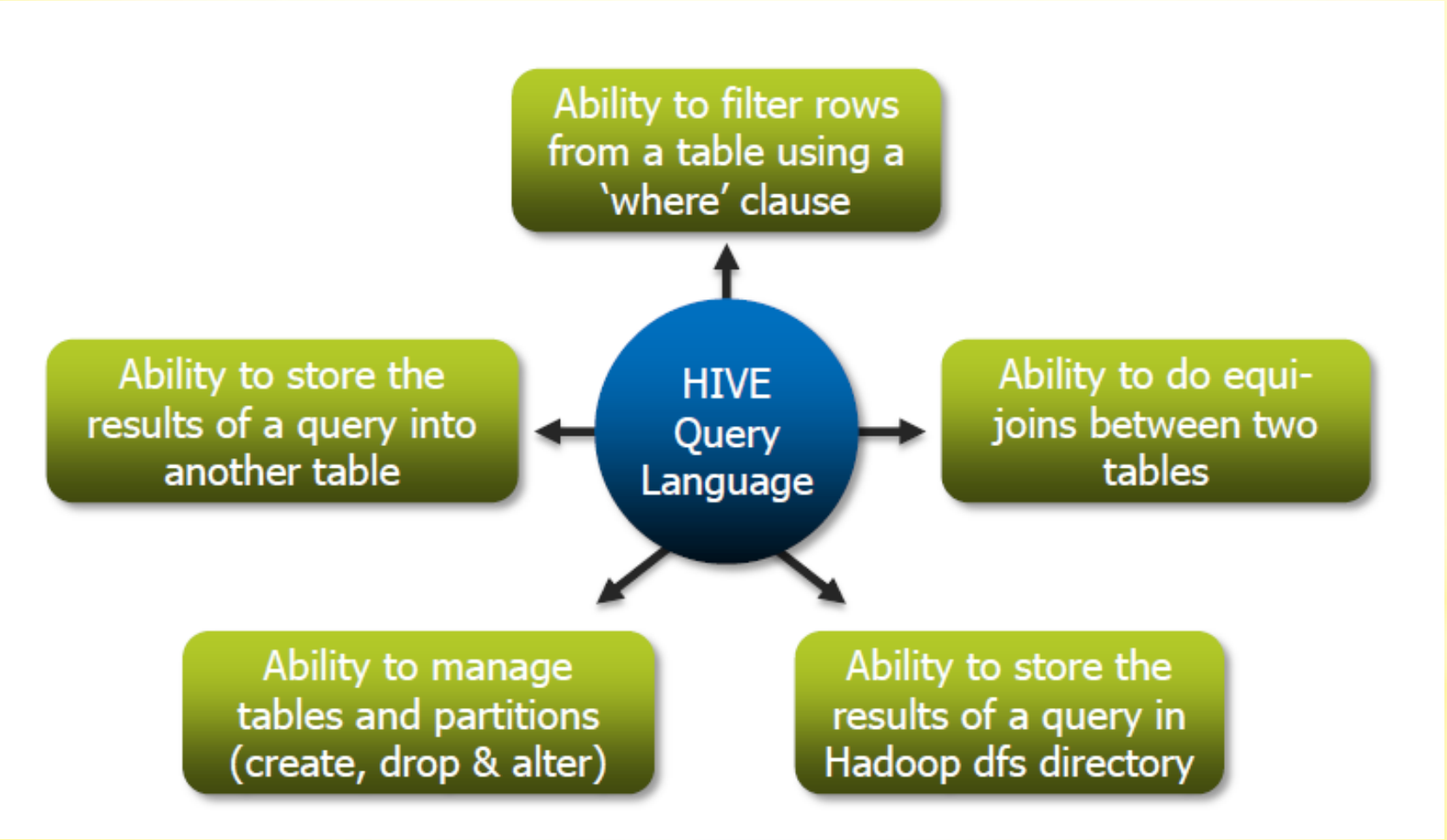
Metastore

HIVE Service JVM

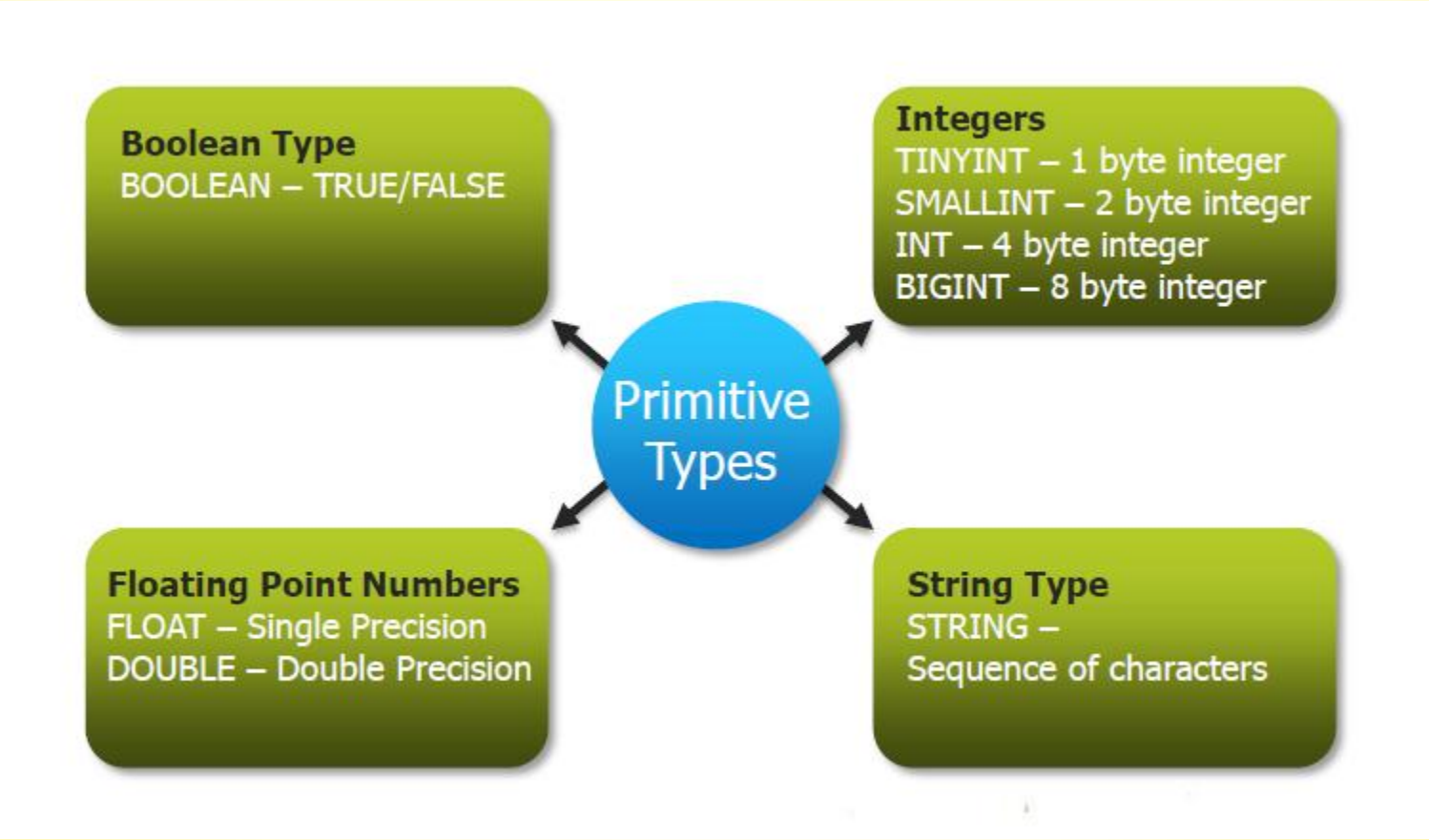


Abilities of Hive Query Language

Hive Query Language provides the basic SQL-like operations

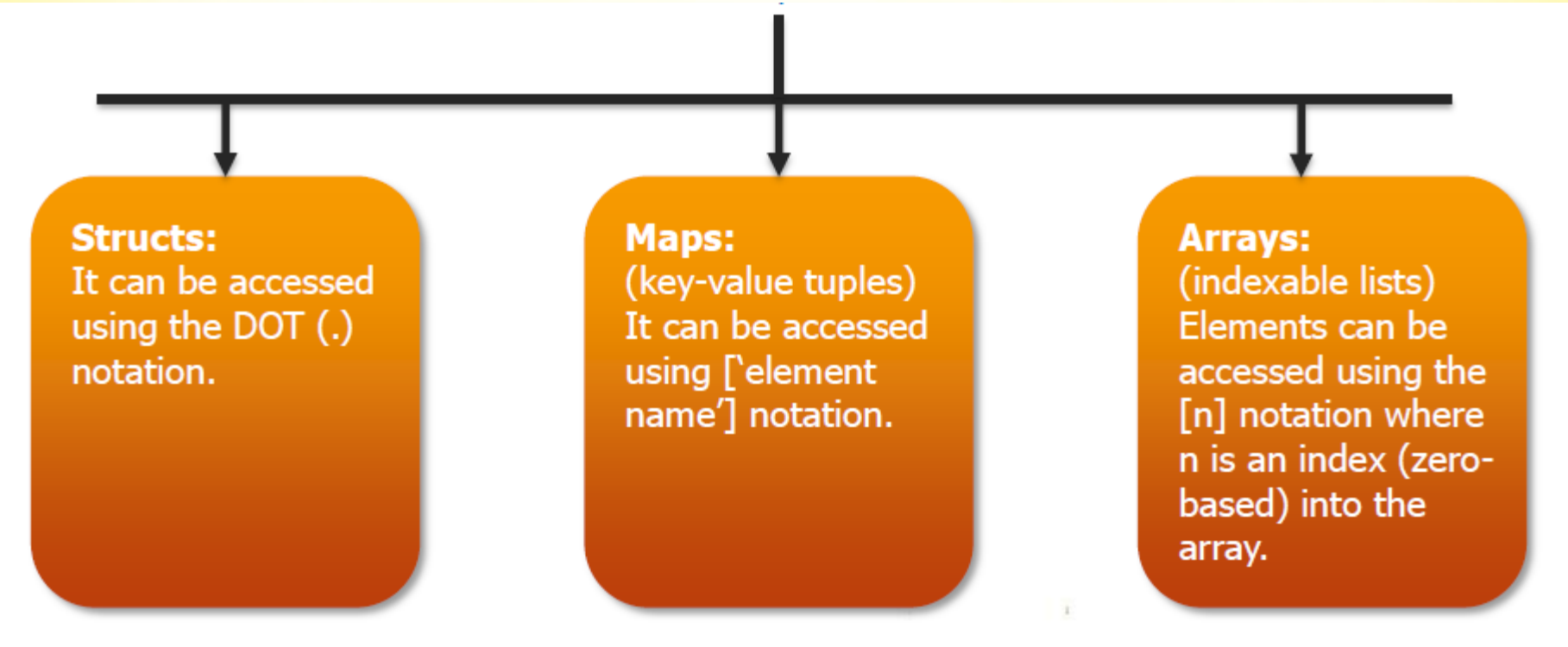


Type System


Boolean Type
BOOLEAN – TRUE/FALSE
Integers
TINYINT – 1 byte integer
SMALLINT – 2 byte integer
INT – 4 byte integer
BIGINT – 8 byte integer
Primitive Types
Floating Point Numbers
FLOAT – Single Precision
DOUBLE – Double Precision
String Type
STRING –
Sequence of characters

Complex Types

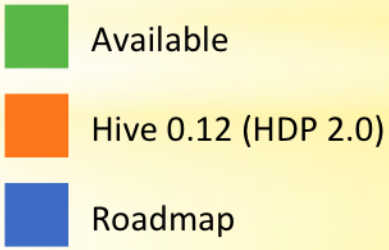
Complex Types can be built up from primitive types and other composite types using the following three operators:



Data Types

Hive SQL Datatypes
INT
TINYINT/SMALLINT/BIGINT
BOOLEAN
FLOAT
DOUBLE
STRING
TIMESTAMP
BINARY
DECIMAL
ARRAY, MAP, STRUCT, UNION
DATE
VARCHAR
CHAR

Hive SQL Semantics
SELECT, INSERT
GROUP BY, ORDER BY, SORT BY
JOIN on explicit join key
Inner, outer, cross and semi joins
Sub-queries in FROM clause
ROLLUP and CUBE
UNION
Windowing Functions (OVER, RANK, etc)
Custom Java UDFs
Standard Aggregation (SUM, AVG, etc.)
Advanced UDFs (ngram, Xpath, URL)
Sub-queries for IN/NOT IN, HAVING
INTERSECT / EXCEPT
Expanded JOIN Syntax



Differences with Traditional RDBMS

- ✓ **Schema on Read vs Schema on Write**
  - ✓ **Hive does not verifies the data when it is loaded**, but rather when a query is issued.
  - ✓ Schema on read makes for a **very fast initial load**, since the data does not have to be read, parsed and serialized to disk in the database's internal format. The load operation is just a file copy or move.
- ✓ **No Updates, Transactions and Indexes**

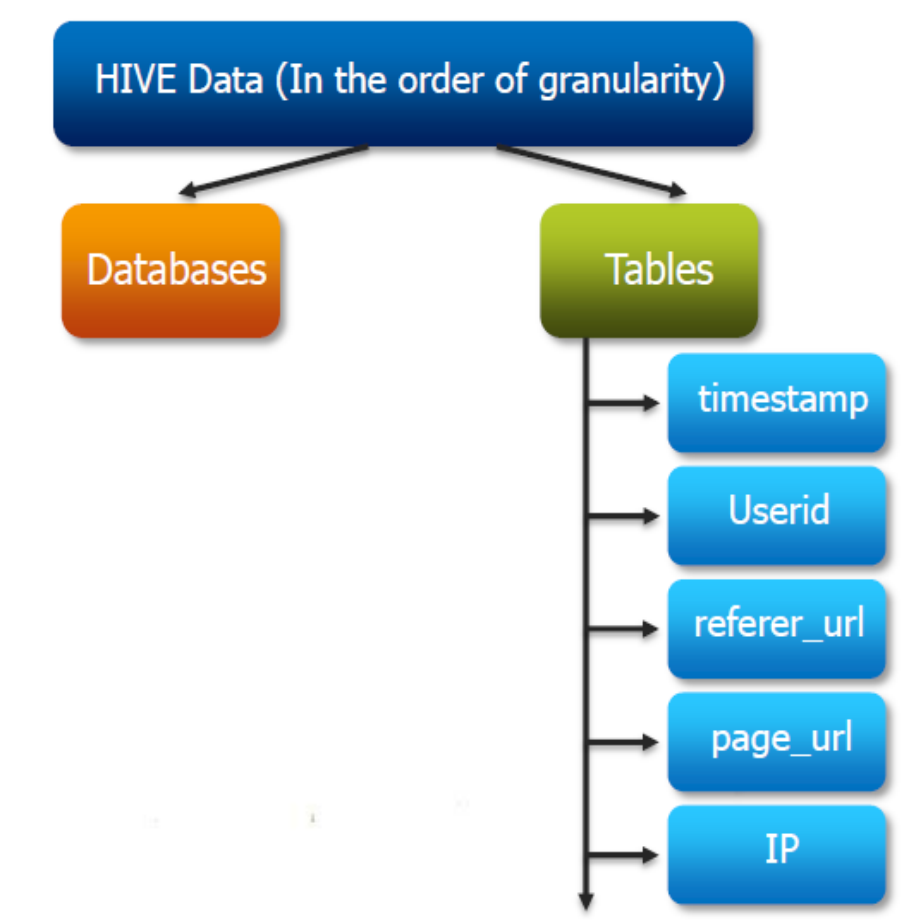
Hive Data Models

Databases
- ✓ Namespaces

Tables
- ✓ Schemas in namespaces

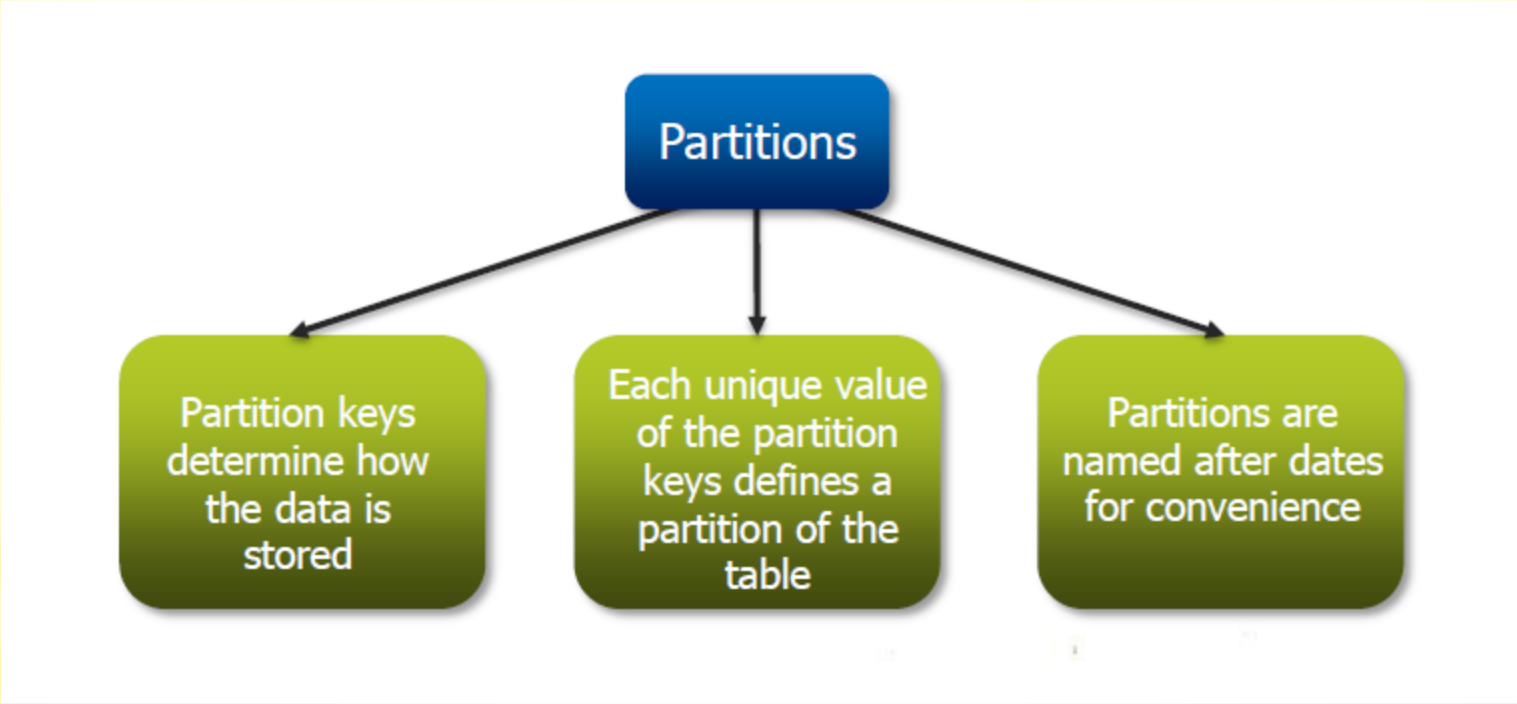
Partitions
- ✓ How data is stored in HDFS
- ✓ Grouping data based on some column
- ✓ Can have one or more columns

Buckets or Clusters
- ✓ Partitions divided further into buckets based on some other column
- ✓ Use for data sampling



Partitions

Partition means dividing a table into a coarse grained parts based on the value of a partition column such as a date. This make it faster to do queries on slices of the data.



Create Database and Table

Create Database

- ✓ Create database retail;

Use Database

- ✓ Use retail;

Create table for storing transactional records

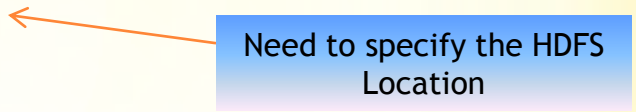
- ✓ Create table txnrecords(txnno INT, txndate STRING, custno INT, amount DOUBLE, category STRING, product STRING, city STRING, state String, Spendby String)
- ✓ Row format delimited
- ✓ Fields terminated by ',' stored as textfile

External Tables

Create the table in another hdfs location and not in warehouse directory

Not managed by hive

- CREATE EXTERNAL TABLE external_Table(dummy STRING)
- LOCATION '/user/notroot/external_table';



Hive does not delete the table (or hdfs files) even when the tables are dropped.

- It leaves the table untouched and only metadata about the tables are deleted

Load Data

- **Load the data into the table**
  - LOAD DATA LOCAL INPATH '/home/ubuntu/notroot/data/txn.csv'
  - OVERWRITE INTO TABLE txnrecords;

- **Describing metadata or schema of the table**
  - Describe txnrecords;

Queries

Select

```
Select count(*)from txnrecords;
```

Aggregation

```
Select count (DISTINCT category) from txnrecords;
```

Grouping

```
Select category, sum( amount) from txnrecords
group by category
```

Managing Outputs

Inserting Output into another table

INSERT OVERWRITE TABLE results (SELECT * from txnrecords)

Inserting into local file

INSERT OVERWRITE LOCAL DIRECTORY 'tmp/results' (SELECT * from txnrecords)

Limitations of Hive