CS523 - BDT

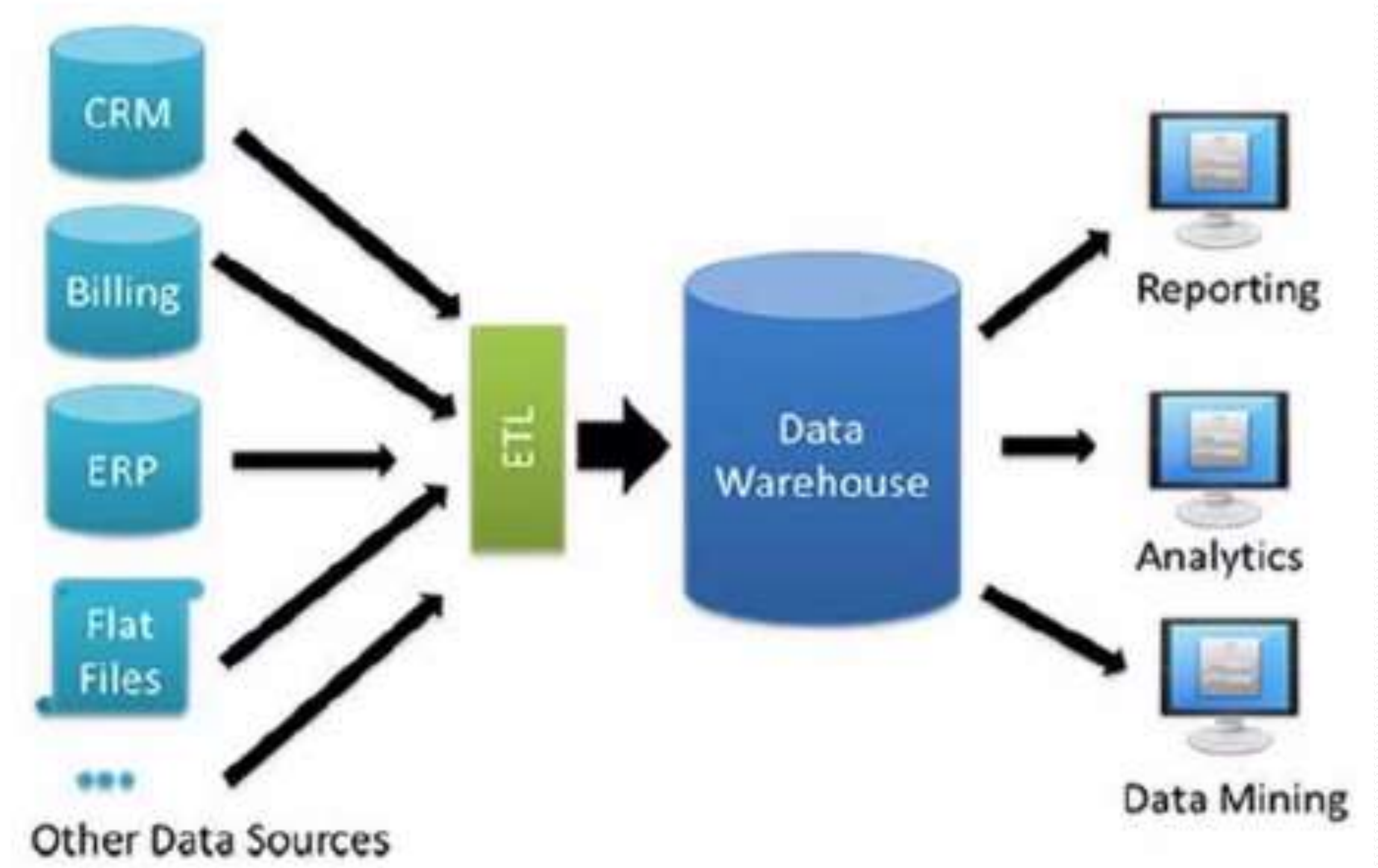
Big Data Technologies

Apache Hive

(Data Warehouse for Hadoop)

Data Warehouse

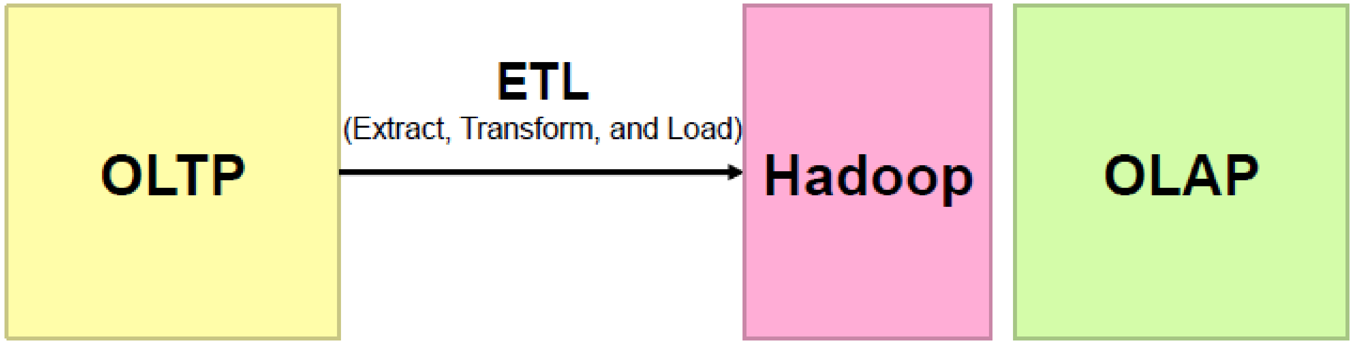
- A large store of data accumulated from a wide range of sources within a company and used to guide management decisions. They provide a SQL interface.



© 2018-19 Maharishi University of Management all rights reserved

Situation before 2006

- At Facebook
  - Data was collected by nightly cron jobs into Oracle
  - "ETL" into D/W systems via hand-coded python
  - Data Grew from 10s of GBs (2006) to 1 TB/day of new data (2007), now 10x of that number.
  - Every day need to fire 70,000 queries on their data



cron is a Linux utility which schedules a command or script on your server to **run** automatically at a specified time and date. A cron job is the scheduled task itself. Cron jobs can be very useful to automate repetitive tasks.

© 2018-19 Maharishi University of Management all rights reserved

Apache Hive

- Developed by Facebook, now open source
- Hive was created to make it possible for analysts with strong SQL skills (but very poor Java skills) to run queries on the huge volumes of data that Facebook stored in HDFS.
- Today, Hive is a successful Apache project used by many organizations as a general-purpose, scalable data processing platform.
- Hive is not a Database, especially in terms of optimizations and it doesn't have it's own storage.
- It enables you to write SQL code which then gets converted to MapReduce programs.
- Of course, SQL isn't ideal for every big data problem—it's not a good fit for building complex machine-learning algorithms, for example—but it's great for many analyses, and it has the huge advantage of being very well known in the industry.
- What's more, SQL is the common language in business intelligence tools (ODBC is a common bridge, for example), so Hive is well placed to integrate with these products.

© 2018-19 Maharishi University of Management all rights reserved

Hive Usage @ Facebook

- **Statistics per day:**
  - 4 TB of compressed new data added per day
  - 135TB of compressed data scanned per day
  - 7500+ Hive jobs on per day
- **Hive simplifies Hadoop:**
  - ~200 people/month run jobs on Hadoop/Hive
  - Analysts (non-engineers) use Hadoop through Hive
  - 95% of jobs are Hive Jobs

© 2018-19 Maharishi University of Management all rights reserved

Apache Hive

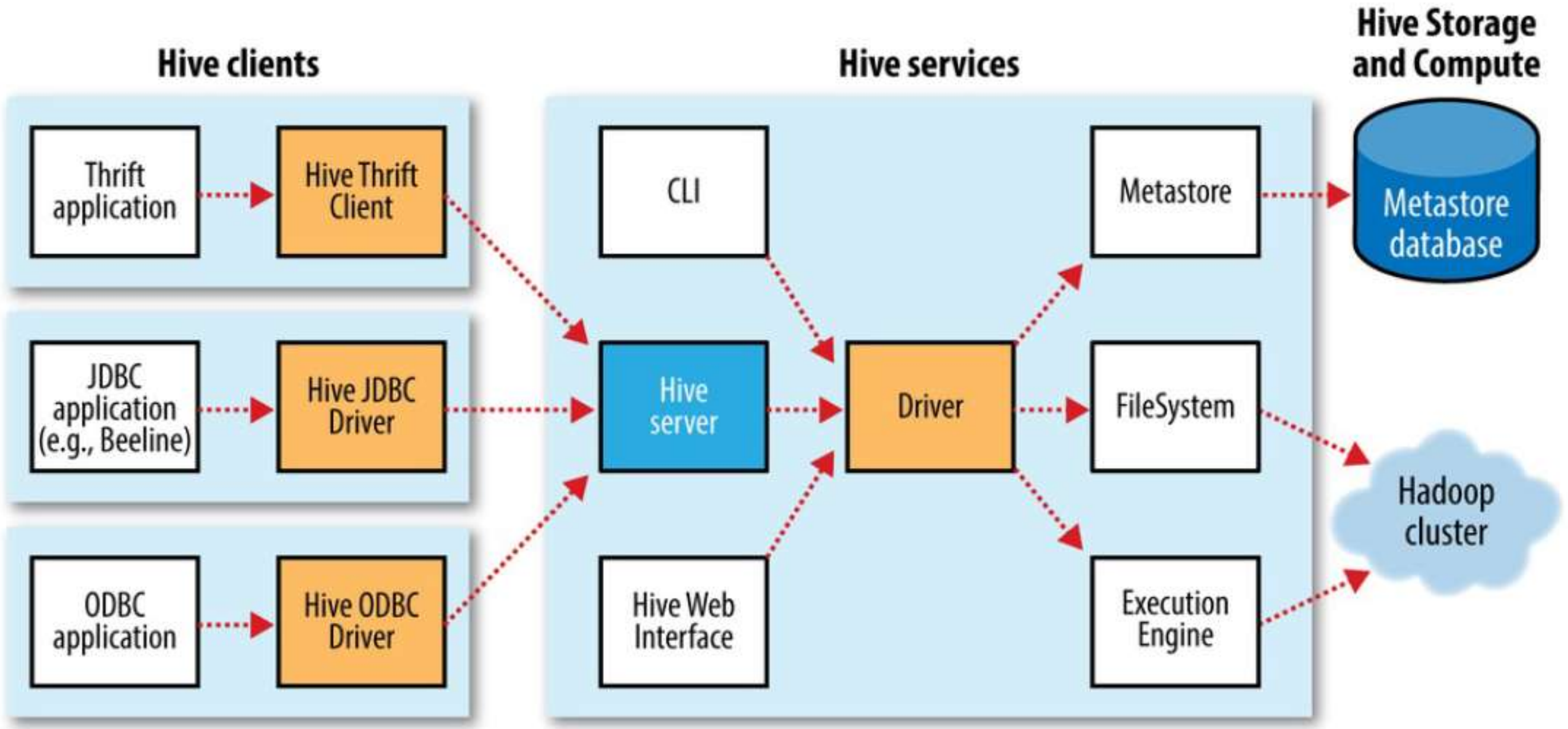
- Data warehouse infrastructure built on top of Hadoop.
- Hive provides a mechanism to project structure onto the data and query the data using SQL-like language called HiveQL.
- Hive uses Mapreduce and HDFS for processing and storage/retrieval of data
  - Tables are stored in HDFS as flat files
  - Can also use Tez or Spark as execution engine
- It stores schema in a database and processed data into HDFS as files.
- In Hive, tables and databases are created first and then data is loaded into these tables.
- **Main Idea:**
  - Provide higher-level language to facilitate large-data processing
  - Higher-level language “compiles down” to MapReduce jobs

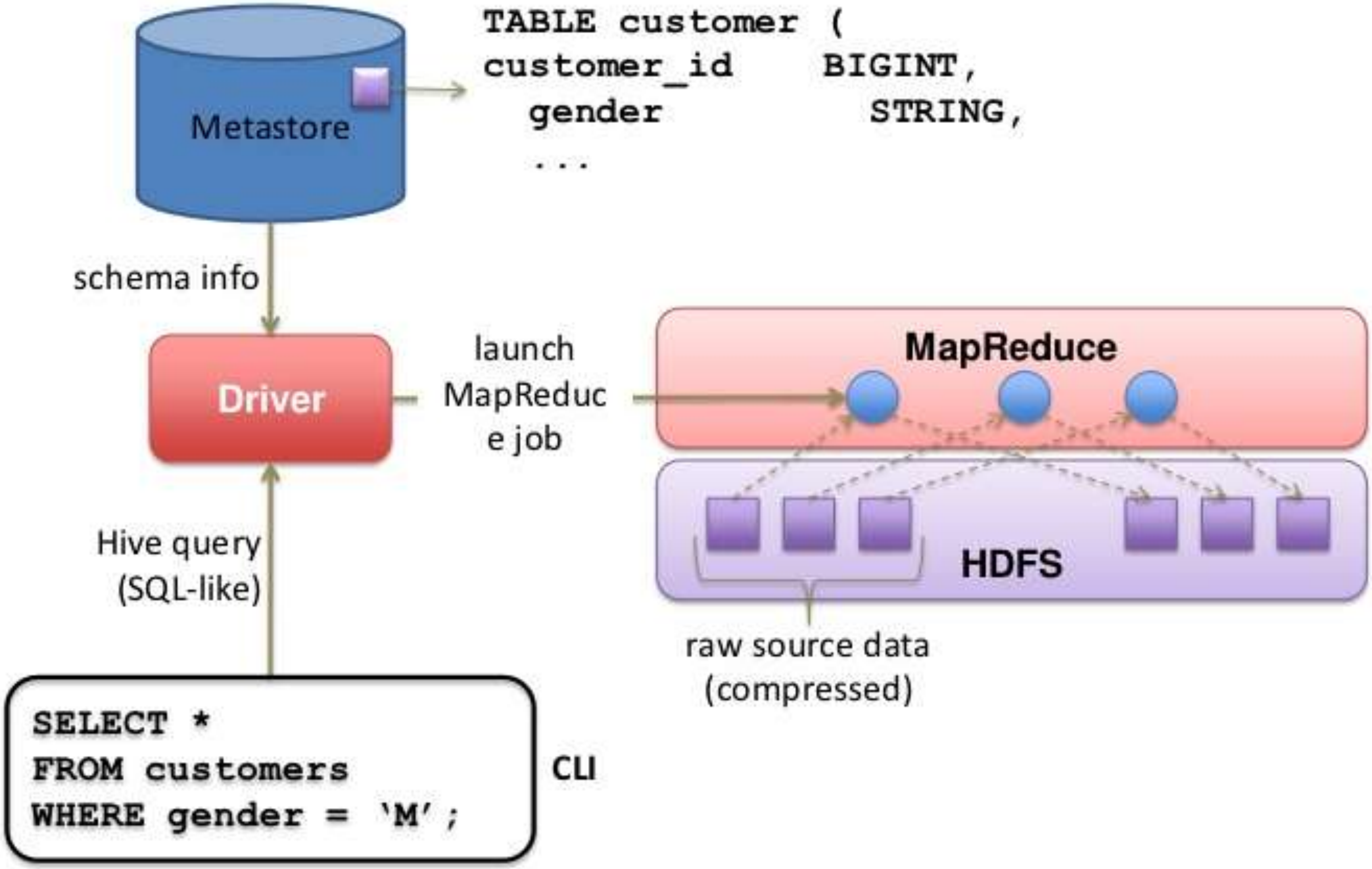
Apache Hive

- **Hive is not**
  - A relational database
  - A design for OnLine Transaction Processing (OLTP)
  - A language for real-time queries and row-level updates
- **Hive is**
  - A system for managing and querying unstructured data as if it were structured!
  - Designed for OLAP
  - Familiar, fast, scalable, and extensible.

© 2018-19 Maharishi University of Management all rights reserved

Hive Architecture





```
TABLE customer (
customer_id    BIGINT,
  gender          STRING,
  ...
```

```
SELECT *
FROM customers
WHERE gender = 'M';
```

Hive Major Components

- **Shell**: Allows interactive queries
- **Driver**: The Driver is used for receiving queries from user and it sends them to Hadoop.
- **Compiler** :- It parses the query, does semantic analysis on the different query blocks and query expressions and eventually generates an execution plan with the help of the table and partition metadata looked up from the metastore.
- **Execution Engine** :- This is the component which executes the execution plan created by the compiler.
  - The plan is a DAG of stages.
  - The execution engine manages the dependencies between these different stages of the plan and executes these stages on the appropriate system components.

Hive Major Components

- **MetaStore** :- The component that maintains metadata about Hive in a relational database.
  - This metadata contains information about what tables exist, their columns, user privileges, the corresponding HDFS files where the data is stored and more.
  - By default, Hive uses Derby, an embedded Java relational database, to store the metadata.

© 2018-19 Maharishi University of Management all rights reserved

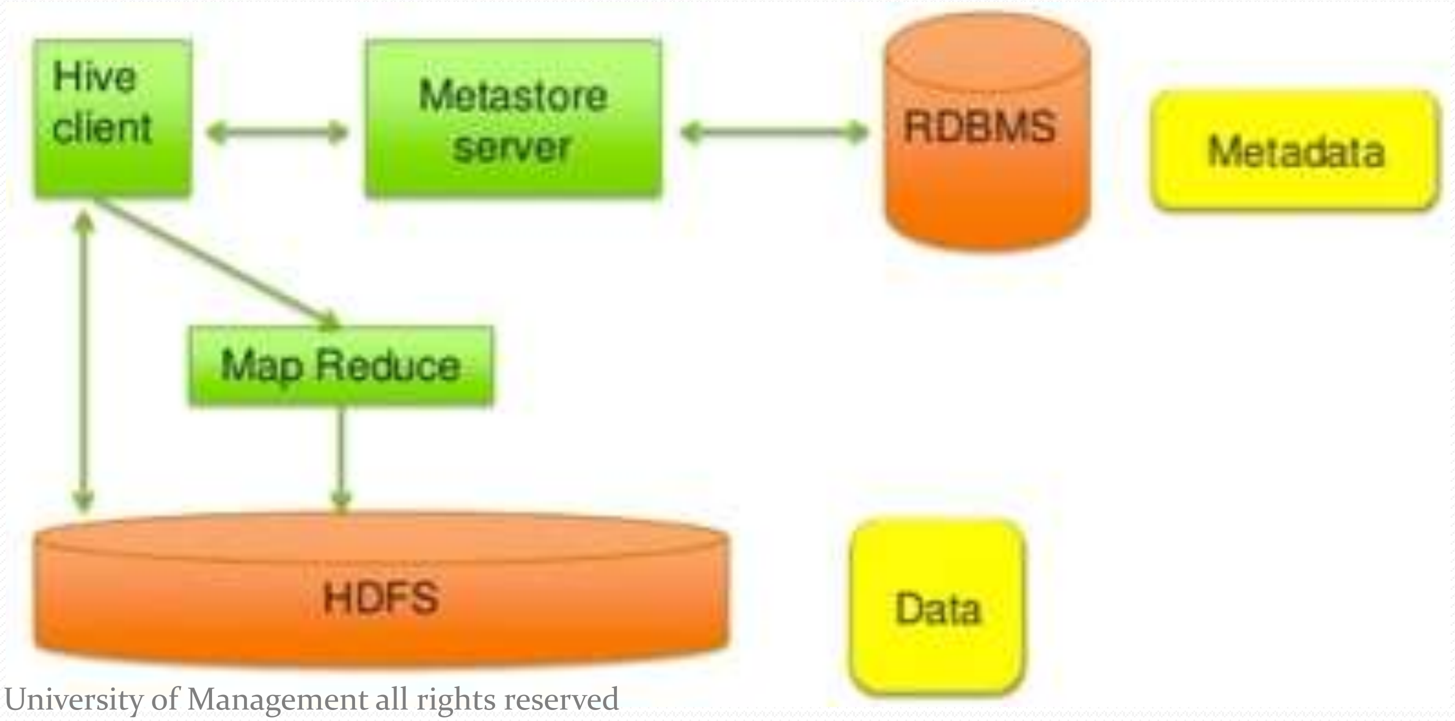
Data Hierarchy

- Hive is organized hierarchically into:
  - **Databases**: namespaces that separate tables and other objects
  - **Tables**: homogeneous units of data with the same schema
    - Analogous to tables in an RDBMS
  - **Partitions**: determine how the data is stored
    - Allow efficient access to subsets of the data
    - For example, range-partition tables by date
  - **Buckets/clusters**
    - For subsampling within a partition
    - Join optimization

© 2018-19 Maharishi University of Management all rights reserved

The Hive Metastore (Metadata Store)

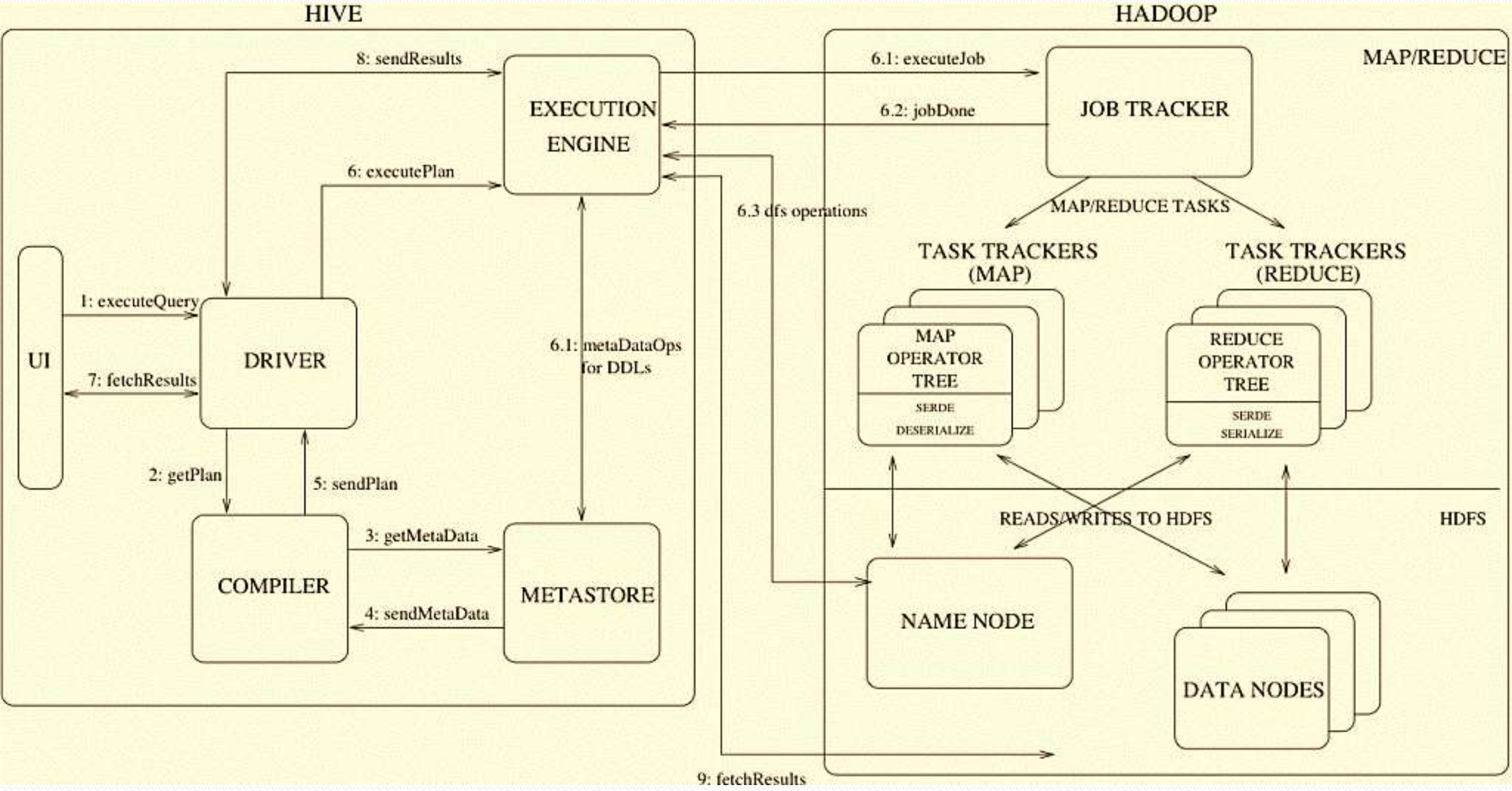
- The *metastore* is the central repository of Hive metadata
- Database: namespace containing a set of tables
- Holds table definitions (column types, physical layout)
- Holds partitioning information
- Can be stored in Derby, MySQL, and many other relational DBs



Hive Warehouse

- **Hive tables are stored in the Hive "warehouse"**
  - E.g., /user/hive/warehouse/students
- **Tables stored in subdirectories of warehouse**
  - Each table has a corresponding HDFS directory
  - Partitions form subdirectories of tables
- **Actual data stored in flat files**
  - Users can associate a table with a custom SerDe format
    - SerDe - Describes how to load the data from the file into a representation that makes it look like a table

Hive In Progress



Hive In Progress

- **Step 1 :-** The UI calls the execute interface to the Driver
- **Step 2 :-** The Driver creates a session handle for the query and sends the query to the compiler to generate an execution plan
- **Step 3 & 4 :-** The compiler needs the metadata and so it sends a request for getMetaData and receives the sendMetaData request from MetaStore.
- **Step 5 :-** This metadata is used to typecheck the expressions in the query tree as well as to prune partitions based on query predicates. The plan generated by the compiler is a DAG of stages with each stage being either a map/reduce job, a metadata operation or an operation on HDFS. For map/reduce stages, the plan contains map operator trees (operator trees that are executed on the mappers) and a reduce operator tree (for operations that need reducers).

© 2018-19 Maharishi University of Management all rights reserved

Hive In Progress

- **Step 6 :-** The execution engine submits these stages to appropriate components (steps 6, 6.1, 6.2 and 6.3).
  - In each task (mapper/reducer) the deserializer associated with the table or intermediate outputs is used to read the rows from HDFS files and these are passed through the associated operator tree.
  - Once the output gets generated it is written to a temporary HDFS file though the serializer. The temporary files are used to provide the subsequent map/reduce stages of the plan. For DML operations the final temporary file is moved to the table's location.
- **Step 7 & 8 & 9 :-** For queries, the contents of the temporary file are read by the execution engine directly from HDFS as part of the fetch call from the Driver.

© 2018-19 Maharishi University of Management all rights reserved

Schema on Read Vs. Schema on Write

- In a traditional database, a table's schema is enforced at data load time.
- If the data being loaded doesn't conform to the schema, then it is rejected. This design is sometimes called ***schema on write*** because the data is checked against the schema when it is written into the database.
- Hive, on the other hand, doesn't verify the data when it is loaded, but rather when a query is issued. This is called ***schema on read***.

© 2018-19 Maharishi University of Management all rights reserved

Hive Table

- A Hive table is a logical concept that’s physically composed of a number of files in HDFS.
- A Hive table is logically made up of the data being stored and the associated metadata describing the layout of the data in the table.
- The data typically resides in HDFS, although it may reside in any Hadoop filesystem, including the local filesystem or S3.
- Hive stores the metadata in a relational database and not in HDFS.
- By default, a created table in Hive is an internal (managed) table.

© 2018-19 Maharishi University of Management all rights reserved

Internal and External Tables

- Tables can either be **internal (managed table)**, where Hive organizes them inside a warehouse directory (controlled by the *hive.metastore.warehouse.dir* property with a default value of */user/hive/warehouse* [in HDFS]), or they can be **external**, in which case Hive doesn't manage them.
- Internal tables are useful if you want Hive to manage the complete lifecycle of your data, including the deletion, whereas external tables are useful when the files are being used outside of Hive.
- Tables can be partitioned, which is a physical arrangement of data, into distinct subdirectories for each unique partitioned key.

© 2018-19 Maharishi University of Management all rights reserved

Internal & External Tables

Managed or Internal Tables

-When location is not defined; tables created in default warehouse directory

-When we drop table, hive deletes data in table

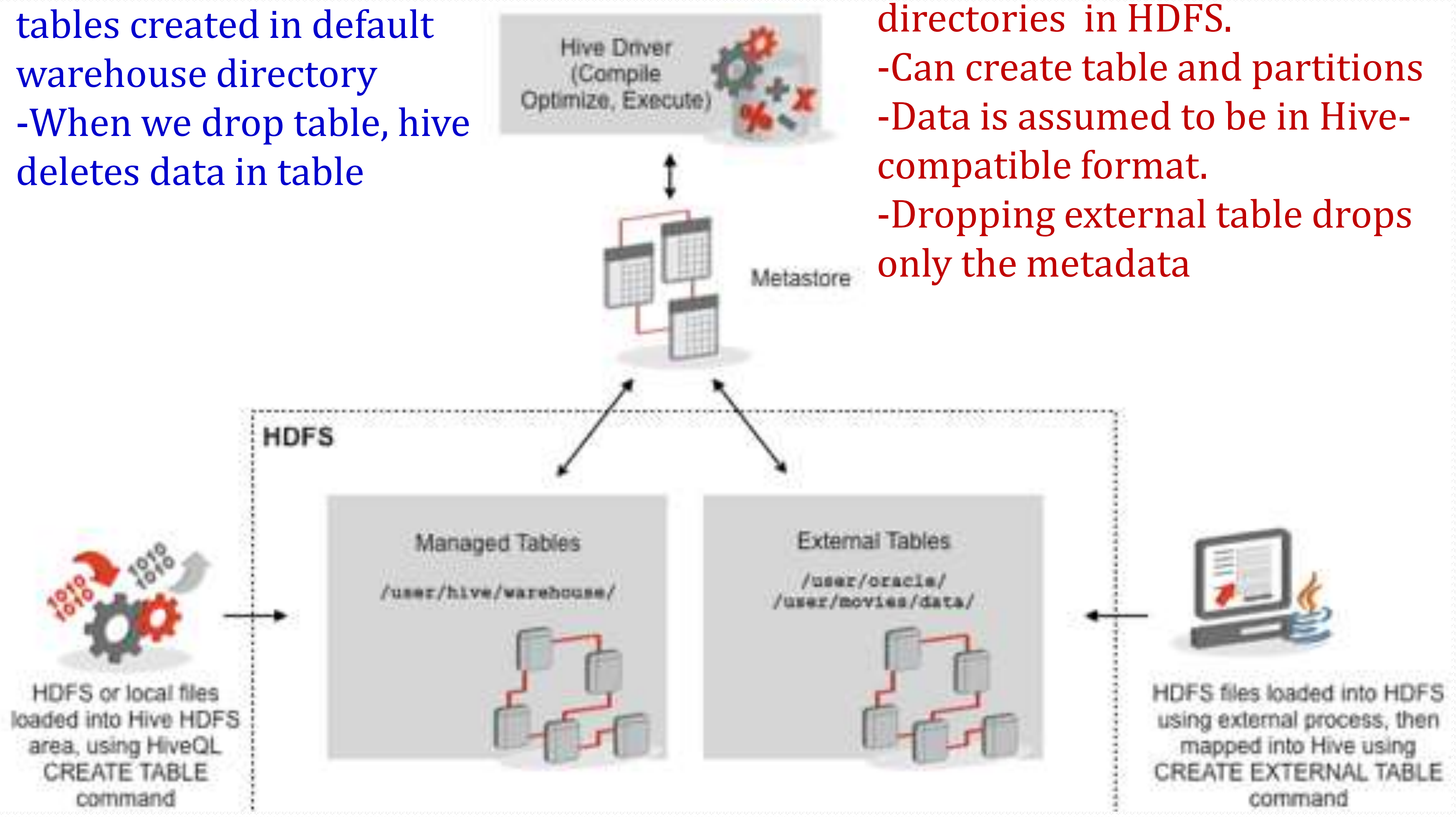
External Tables

-Point to existing data directories in HDFS.

-Can create table and partitions

-Data is assumed to be in Hive-compatible format.

-Dropping external table drops only the metadata


Hive Driver (Compile Optimize, Execute)
Metastore
HDFS
Managed Tables
/user/hive/warehouse/
External Tables
/user/oracle/
/user/movies/data/
HDFS or local files loaded into Hive HDFS area, using HiveQL CREATE TABLE command
HDFS files loaded into HDFS using external process, then mapped into Hive using CREATE EXTERNAL TABLE command

HiveQL

- HiveQL is Hive's query language, a dialect of SQL. It is heavily influenced by MySQL.
- HiveQL / HQL provides the basic SQL-like operations:
  - Select columns using SELECT
  - Filter rows using WHERE
  - JOIN between tables
  - Evaluate aggregates using GROUP BY
  - Store query results into another table
  - Download results to a local directory (i.e., export from HDFS)
  - Manage tables and queries with CREATE, DROP, and ALTER

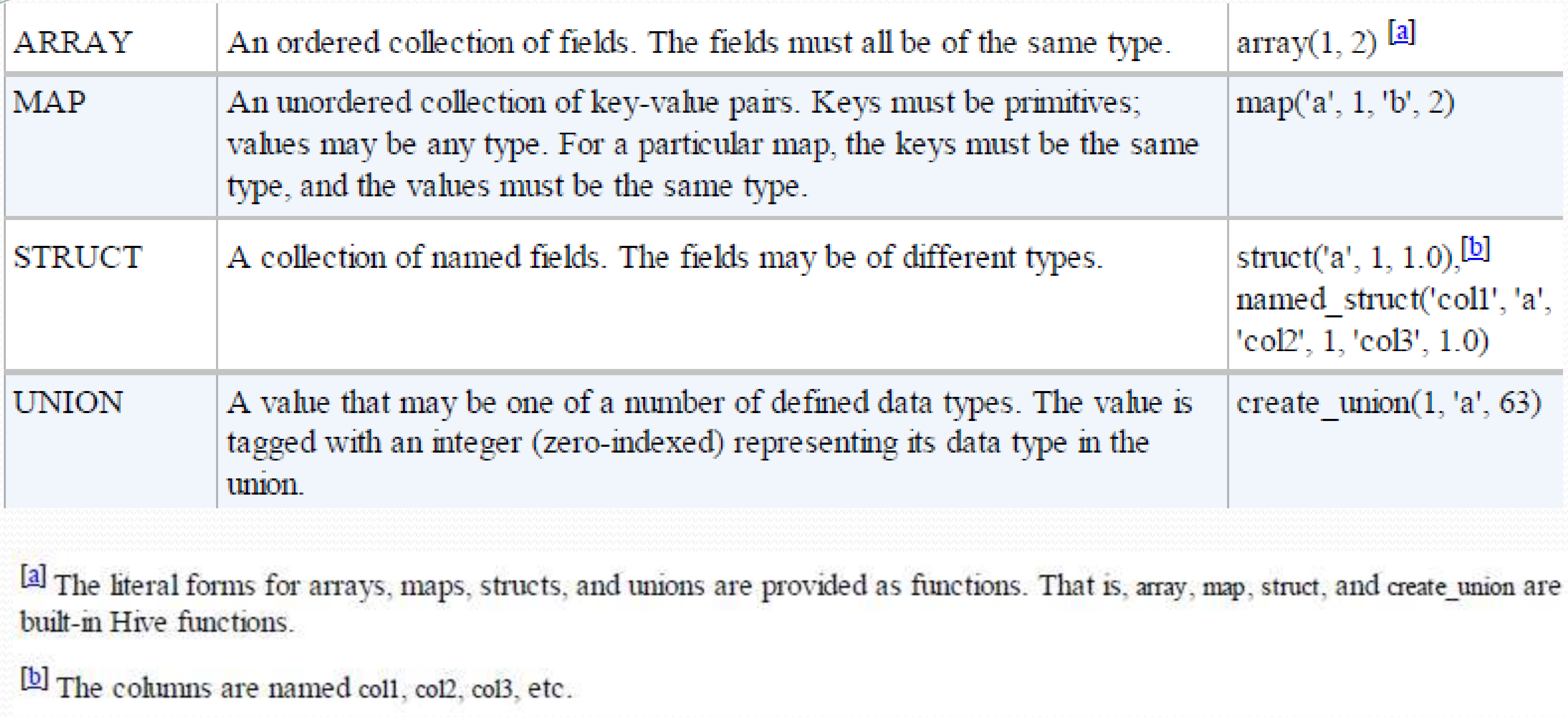
© 2018-19 Maharishi University of Management all rights reserved

Primitive Data Types in Hive

Type	Description	Literal examples
BOOLEAN	True/false value.	TRUE
SMALLINT	2-byte (16-bit) signed integer, from –32,768 to 32,767.	1S
INT	4-byte (32-bit) signed integer, from –2,147,483,648 to 2,147,483,647.	1
BIGINT	8-byte (64-bit) signed integer, from –9,223,372,036,854,775,808 to 9,223,372,036,854,775,807.	1L
FLOAT	4-byte (32-bit) single-precision floating-point number.	1.0
DOUBLE	8-byte (64-bit) double-precision floating-point number.	1.0
DECIMAL	Arbitrary-precision signed decimal number.	1.0
STRING	Unbounded variable-length character string.	'a', "a"
VARCHAR	Variable-length character string.	'a', "a"
CHAR	Fixed-length character string.	'a', "a"
BINARY	Byte array.	Not supported
TIMESTAMP	Timestamp with nanosecond precision.	1325502245000, '2012-01-02 03:04:05.123456789'
DATE	Date.	'2012-01-02'

© 2018-19 Maharishi University of Management all rights reserved

Complex Data Types in Hive



ARRAY	An ordered collection of fields. The fields must all be of the same type.	array(1, 2) [a]
MAP	An unordered collection of key-value pairs. Keys must be primitives; values may be any type. For a particular map, the keys must be the same type, and the values must be the same type.	map('a', 1, 'b', 2)
STRUCT	A collection of named fields. The fields may be of different types.	struct('a', 1, 1.0),[b] named_struct('col1', 'a', 'col2', 1, 'col3', 1.0)
UNION	A value that may be one of a number of defined data types. The value is tagged with an integer (zero-indexed) representing its data type in the union.	create_union(1, 'a', 63)

[a] The literal forms for arrays, maps, structs, and unions are provided as functions. That is, array, map, struct, and create_union are built-in Hive functions.

[b] The columns are named col1, col2, col3, etc.

The Hive Shell

- The shell is the primary way that we will interact with Hive, by issuing commands in HiveQL.
- Open new terminal and fire up hive by just typing hive.
- When starting Hive for the first time, we can check that it is working by listing its tables—there should be none.
- The command must be terminated with a semicolon to tell Hive to execute it:

```
hive> SHOW TABLES;
OK
Time taken: 0.473 seconds
```

© 2018-19 Maharishi University of Management all rights reserved

HQL Example

- hive> show tables;
- hive> CREATE TABLE shakespeare (word STRING, freq INT)

  ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t'

  STORED AS TEXTFILE
- hive> LOAD DATA LOCAL INPATH 'shakespeareWrdFqy.txt'

  INTO TABLE shakespeare;
- hive> SELECT * FROM shakespeare

  WHERE freq>100

  SORT BY freq ASC

  LIMIT 10;

© 2018-19 Maharishi University of Management all rights reserved

Creating Schema for Hive Table: Create Table Syntax

```
CREATE TABLE employees (
     name         STRING,
     salary       FLOAT,
     dept         STRING
     )
ROW FORMAT
     DELIMITED
          FIELDS TERMINATED BY ‘,'
          LINES TERMINATED BY '\n';
```

- The **ROW FORMAT** clause, is particular to HiveQL. This declaration is saying that each field is separated by “comma” and each row in the data file is separated by new line.
- **The above query will create an internal table.**

© 2018-19 Maharishi University of Management all rights reserved

Creating Schema for Hive Table: Create Table Syntax

- By default, tables are assumed to be of text input format,
- Default field delimiters are ^A(ctrl-a) –\u0001
- Default record delimiter is – \n
- **Custom SerDe** - Depending on the nature of data the user has, the inbuilt SerDe may not satisfy the format of the data. So users need to use custom SERDE with SERDEPROPERTIES.

© 2018-19 Maharishi University of Management all rights reserved

Table for Apache Weblog Data

```
CREATE TABLE apachelog (
  host STRING,
  identity STRING,
  user STRING,
  time STRING,
  request STRING,
  status STRING,
  size STRING,
  referer STRING,
  agent STRING)
ROW FORMAT SERDE
       'org.apache.hadoop.hive.serde2.RegexSerDe'
WITH SERDEPROPERTIES (
  "input.regex" = "([^]*) ([^]*) ([^]*) (-|\\[^\\]*\\])
([^ \"]*|\"[^\"]*\") (-|[0-9]*) (-|[0-9]*)(?: ([^
\"]*|\".*\") ([^ \"]*|\".*\"))?"
)
STORED AS TEXTFILE;
```

Creating a Complex Schema for Hive Table

```
CREATE TABLE employees (
      name  STRING,
      salary      FLOAT,
      department  STRING
      subordinates      ARRAY<STRING>,
      deductions  MAP<STRING, FLOAT>,
      address     STRUCT<street:STRING, city:STRING,
                              state:STRING, zip:INT>)
COMMENT 'This is the page view table'
PARTITIONED BY(department STRING)
ROW FORMAT DELIMETED FIELDS TERMINATED BY '\t'
COLLECTION ITEMS TERMINATED BY ','
MAP KEYS TERMINATED BY '#'
LINES TERMINATED BY '\n'
STORED AS TEXTFILE;
```

© 2018-19 Maharishi University of Management all rights reserved

Loading Local Data into Hive Table

```
LOAD DATA LOCAL INPATH
'/home/cloudera/cs523/input/employee.txt'
INTO TABLE employees;
```

- Running this command tells Hive to **copy** the specified local file in its warehouse directory (by default).
- The default location of Hive table can be overwritten by using LOCATION.
- '**LOCAL**' signifies that the input file is on the local file system.
- If 'LOCAL' is omitted then it looks for the file in HDFS.
  - In this case, the HDFS file will be **moved** to the warehouse (by default)

© 2018-19 Maharishi University of Management all rights reserved

Loading HDFS Data into Hive Table

```
LOAD DATA INPATH
'/home/cloudera/cs523/input/employee.txt'
OVERWRITE INTO TABLE employees;
```

- When HDFS data is loaded into the table, the input files are moved to the table location directory. The files are not examined or checked for errors until they are used in a query. (NO verification of data against the schema is performed by the load command.)
- The OVERWRITE keyword in the LOAD DATA statement tells Hive to delete any existing files in the directory for the table.
  - If it is omitted, the new files are simply added to the table's directory (unless they have the same names, in which case they replace the old files).

© 2018-19 Maharishi University of Management all rights reserved

Create External Table

```
CREATE EXTERNAL TABLE employees (
     name STRING,
     salary    FLOAT,
     departmentSTRING
     )
ROW FORMAT
     DELIMETED
          FIELDS TERMINATED BY `,'
          LINES TERMINATED BY '\n';
```

- After deleting the External table only the meta data related to table is deleted but not the contents of the table.
- The above approach will move/copy your data to /user/hive/warehouse directory.
- If you don't want your data to get moved, then you need to mention the external location of the data while creating the table itself as shown next.

© 2018-19 Maharishi University of Management all rights reserved

Create External Table

```
CREATE EXTERNAL TABLE employees (
     name STRING,
     salary    FLOAT,
     departmentSTRING
     )
ROW FORMAT
     DELIMETED
          FIELDS TERMINATED BY ‘,'
          LINES TERMINATED BY '\n';
LOCATION '/user/cloudera/cs523';
```

- Here we have specified the location of the data in the table creation itself. Now if you delete the table, nothing will happen to the data. Data will be intact!
- Also with this LOCATION clause added with the actual file path, do not LOAD the data explicitly!

© 2018-19 Maharishi University of Management all rights reserved

External Tables

- The EXTERNAL keyword in the "create table" statement tells Hive that this table is external and the LOCATION … clause will tell Hive where it is located.
- Because it's external, Hive does not assume it owns the data. Therefore, dropping the table does not delete the data, although the metadata for the table will be deleted.
- You can tell whether or not a table is managed or external using the output of
  `DESCRIBE EXTENDED <tablename> or`
  `DESCRIBE FORMATTED <tablename>;`
  command.

© 2018-19 Maharishi University of Management all rights reserved

When to use
Internal & External tables?

- **Internal table**
  - Data is temporary
  - Hive to Manage the table data completely not allowing any external source to use the table
  - Don't want data after deletion
- **External table**
  - The data is also used outside of Hive. For example, the data files are read and processed by an existing program that doesn't lock the files
  - Hive should not own data and control settings, etc., you have another program or process that will do those things
  - Can create table back with the same schema and point to the location of the data

© 2018-19 Maharishi University of Management all rights reserved

Select * From Table

- In Hive if you do simple query like **select * from table,** no map reduce job is going to run as we are just dumping the data.

- You can add **explain** before your query and it will display how the query is going to be executed by the execution engine and displays how many map reduce phases are going to be required for the query.

- When ever you do **aggregations** then the **reducer phase** will be executed along with map phase.

© 2018-19 Maharishi University of Management all rights reserved

Some more Table Commands

- **Drop table**

```
DROP TABLE tableName;
```

- **Alter table**

```
ALTER TABLE tableName RENAME tableNameNew;
ALTER TABLE tableName ADD COLUMNS (new_col
INT);
```

Relational Operators

- **ALL and DISTINCT**
  - Specify whether duplicate rows should be returned
  - ALL is the default (all matching rows are returned)
  - DISTINCT removes duplicate rows from the result set
- **WHERE**
  - Filters by expression
  - Does not support IN, EXISTS or sub-queries in the WHERE clause
- **LIMIT**
  - Indicates the number of rows to be returned

Relational Operators

- **GROUP BY**
  - Group data by column values
  - Select statement can only include columns included in the GROUP BY clause
- **ORDER BY / SORT BY**
  - ORDER BY performs total ordering
    - Slow, poor performance
  - SORT BY performs partial ordering
    - Sorts output from each reducer

© 2018-19 Maharishi University of Management all rights reserved

Word Count in Hive

```
CREATE TABLE WordCount (line string);


LOAD DATA INPATH '/user/cloudera/test.txt' INTO
TABLE wordcount;


SELECT each_word, count(*) AS count
FROM (
     SELECT explode(split(line, ' ‘)) AS each_word
     FROM wordcount) resultTable
GROUP BY each_word;
```

----***explode()*** takes in an ***array*** (or a map) as an ***input*** and gives the elements of the array (map) as ***separate rows*** for ***output***.

```
hive -f /home/cloudera/cs523/Examples/Hive/wc.sql
```

© 2018-19 Maharishi University of Management all rights reserved

JOIN Example

- Example: Relational join on two tables:
  - Table of word counts from Shakespeare collection
  - Table of word counts from the HTDG

```
SELECT s.word, s.freq, h.freq FROM shakespeare s
  JOIN htdg h ON (s.word = h.word) WHERE s.freq >= 1
                                    AND h.freq >= 1
  ORDER BY s.freq DESC LIMIT 8;

the     25848 62394
I       23031 8854
and     19671 38985
to      18038 13526
of      16700 34654
a       14170 8057
you     12702 2720
my      11297 4135
```

Store Query Results into File

- The following command outputs the table to local directory

```
INSERT OVERWRITE LOCAL DIRECTORY '<directory>'
SELECT * FROM table_name;
```

- The following command outputs the table to an HDFS file

```
INSERT OVERWRITE DIRECTORY
'/user/cloudera/cs523’
SELECT a.* FROM table_name;
```

By default, Hive generate file names as 000000_0 in TSV format.

© 2018-19 Maharishi University of Management all rights reserved

Partitions

- By default, a simple query in Hive scans the whole Hive table. This slows down the performance when querying a large-size table.
- The issue could be resolved by creating Hive partitions, which is very similar to what's in the RDBMS.
- In Hive, each partition corresponds to a predefined partition column(s) and stores it as a subdirectory in the table's directory in HDFS.
- When the table gets queried, only the required partitions (directory) of data in the table are queried, so the I/O and time of query is greatly reduced.
- It is very easy to implement Hive partitions when the table is created.

© 2018-19 Maharishi University of Management all rights reserved

Partitions

- Each table can have one or more partitions which determine the distribution of data within sub-directories of the table directory.
- Partition keys are basic elements for determining how the data is stored in the table.
  - Can make some queries faster
  - Divide data based on partition column
  - Use PARTITION BY clause when creating table
  - Use PARTITION clause when loading data
  - SHOW PARTITIONS will show a table's partitions

© 2018-19 Maharishi University of Management all rights reserved

Partitions

- Suppose data for table employee is in the directory /warehouse/employee.
- If employee is partitioned on column state, then data with a particular state value 'IA' will be stored in files within the directory /warehouse/employee/state=IA.

```
CREATE TABLE employees
     (name STRING, salary FLOAT, dept STRING)
PARTITIONED BY (state string)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ',';

LOAD DATA INPATH
'/user/cloudera/cs523/input/employeeIA.txt'
OVERWRITE INTO TABLE employees
PARTITION (state=IA');
```

© 2018-19 Maharishi University of Management all rights reserved

Hive is not suitable for

- Those cases where latency is very important.
- Those cases when you need to do CRUD operations.
- If we don’t need schema or bringing in schema is difficult or not possible on the data at hand.
- Very small data sets
- If RDBMS can solve the problem, don’t invest in Hive.