CISC 886- Cloud and Big Data

Anwar Hossain, Ph.D.
Queen’s University
Email:
ahossain@queensu.ca
anwar.Hossain@gmail.com

Agenda

- HIVE

Intro

- HIVE → to execute SQL like command and in the background mapreduce jobs are executed.
- Data warehousing solutions for big data on Hadoop
- Developed by facebook
- Select, avg etc SQL command
- Hive takes the SQL commands and converts them into MapReduce jobs
- particularly designed for online analytical processing systems (OLAP).
- Particularly suitable for data summarization, data querying and data analysis

HIVE limitations

- Limited indexing capability
- Very High level transaction support -- ACID
- No triggers
- Should not be considered as DB
- Hive is not a relational database or an architecture for online transaction processing (OLTP).

HIVE vs DBMS

- DB→ schema on write

Col1/ field1	Col2/ field2	Col3	Col4

- HIVE→ schema on read
  - Data is not verified when loaded in Hive, rather when a query is applied
  - The load is simply a file copy or move

HIVE commands

- Create database testdb;
- Use testdb;
- Database→ /user/hive/warehouse/dbname.db
- Table→ /user/hive/warehouse/dbname.db/tablename

Create table

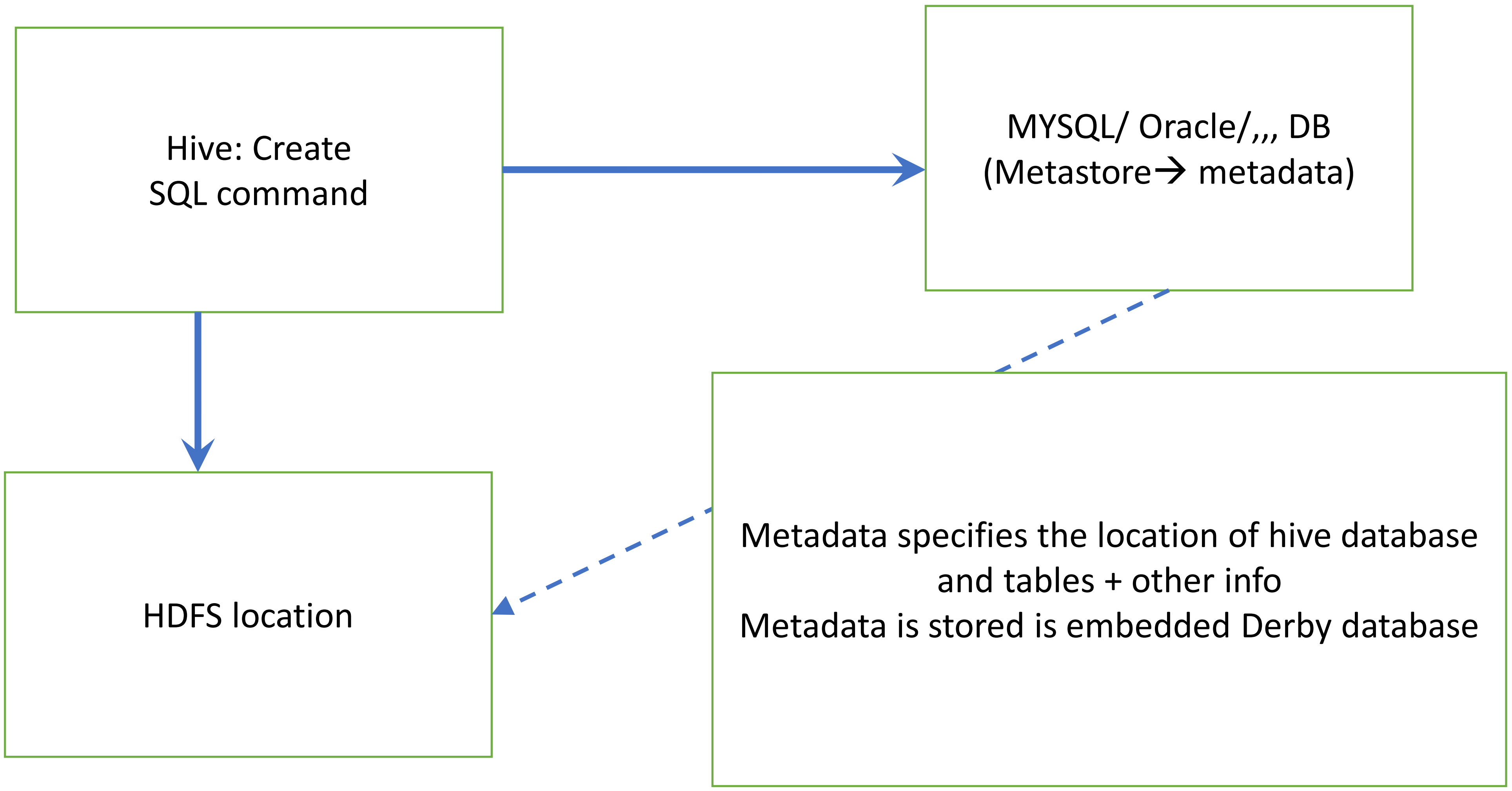
```
hive> create external table if not exists stockstb (
        sym string,
        ymd string,
        priceopen float,
        pricehigh float,
        pricelow float,
        priceclose float,
        priceadjclose float,
        vol int)
        row format delimited
        fields terminated by ','
        location '/user/osboxes/stocks’;
```

1) Managed – hive is the only app that is accessing the data/ if you drop a table, all data of gone
2) External – other application also sharing/ if you drop a table, the data still there

Describe/ Select

- Describe formatted stocktb;
- Select * from stockstb limit 100;

Hive table metadata



Hive: Create
SQL command

MYSQL/ Oracle/,,, DB
(Metastore→ metadata)

HDFS location

Metadata specifies the location of hive database and tables + other info
Metadata is stored is embedded Derby database

Data loading in Hive tables

- Loading data after creating table
  1. LOAD command➔ load data inpath ‘hdfspath’ into table tablename
     → moves the data from hdfs location to the specified table location
     → /user/hive/warehouse/dbname/tablename

  2. CTAS – create as select:
     Create table stocks_copy
     AS
     Select * from stocks1

3. Insert…Select:

Appending:

```
Insert into table stocks_copy
Select s.* from stocks1 s;
```

Overwriting:

```
Insert overwrite table stocks_copy
Select s.* from stocks1 s;
```

Using location to load data

```
4. create external table if not exists stockstb (
     sym string,
     ymd string,
     priceopen float,
     pricehigh float,
     pricelow float,
     priceclose float,
     priceadjclose float,
     vol int)
     row format delimited
     fields terminated by ','
     location '/user/osboxes/stocks';
```

Other hive commands

- Select * from stocks1

  Where sym =‘MSFT’;

- Select * from stocks1

  Where sym in (‘MSFT’, ‘TSLA’)

- Select * from stocks1

  Where sym LIKE ‘MSF%’ AND sym RLIKE ‘B.B’;

- Select distinct syml from stocks;

More..

-
  ```
  Select sym, priceopen, priceclose, vol
  CASE
  When vol <20000 then ‘LOW’
  When vol>= 20000 and vol<40000 then ‘MODEST’
  When vol>40000 and vol<60000 then ‘HIGH’
  ELSE  ‘VERY HIGH’
  End as vol_level from stocks1 WHERE sym =‘TSLA’;
  ```

Group

- Select year(ymd), sym, avg(vol) from stocks group by year(ymd), sym;

- Select year(ymd), sym, avg(vol) from stocks group by year(ymd), sym
Having avg(vol)>2000;

Saving output to a file

- Save output to local folder:

```
Insert overwrite local directory ‘/home/osboxes/data/hive/stocks’
    Row format delimited fields terminated by ‘ , ‘
    Select distinct syml from stocks;
    Select year(ymd), sym, avg(vol) from stocks group by year(ymd), sym;
```

- Save output to local folder:

```
Insert overwrite directory ‘data/hive/stocks’
    Row format delimited fields terminated by ‘ , ‘
    Select distinct syml from stocks;
    Select year(ymd), sym, avg(vol) from stocks group by year(ymd), sym;
```

HIVE data units

- Partitions
  - Each table can have one or more partitions identified by partition key
  - Data for a particular partition is located in tablelocation/partitionkey directory in hdfs.
  - Alter table stocks add if not exists

  Partition (sym = ‘TSLA’) location ‘/out/hive/stocks_tsla’;

- Buckets
  - Data in each partition can be divided into buckets based on the hash of a column in a table. Each bucket is stored as a file in the partition directory. Good for sampling and JOIN optimization.

Bucket

```
create external table if not exists stockstb_bucket (
sym string, ymd string,
priceopen float, pricehigh float,
pricelow float, priceclose float,
priceadjclose float, vol int)
Partitioned by (sym string, yr string)
Clustered by (sym) into 5 buckets
row format delimited fields terminated by ‘,’
```

```
Hive> set hive.exec.dynamic.partition = true; ✓
Hive> set hive.exec.max.dynamic.partitions =1000; ✓
Hive> set hive.exec.max.dynamic.partitions.pernode =500; ✓
Hive> set hive.enforce.bucketing=true; ✓
```

Insert after setting the bucketing property

- Insert overwrite table stocks_bucket

  Partition (sym = ‘ABC’ , yr)

  Select *, year(ymd)

  From stocks where year(ymd) in (‘2001’, ‘2002, ‘2003’) and symbol like ‘B%’;

Multiple partitions

```
From stocks s
Insert overwrite table stockspartitions
Partition(sym = ‘TSLA’)
Select * where s.sym = ‘TSLA’
Insert overwrite table stockspartitions
Partition(sym = ‘MSFT)
Select * where s.sym = ‘MSFT’;
```

Dropping table/partition

- Alter table stockspartition

  Drop if exists partition (sym = ‘TSLA’);

HIVE QL DDL

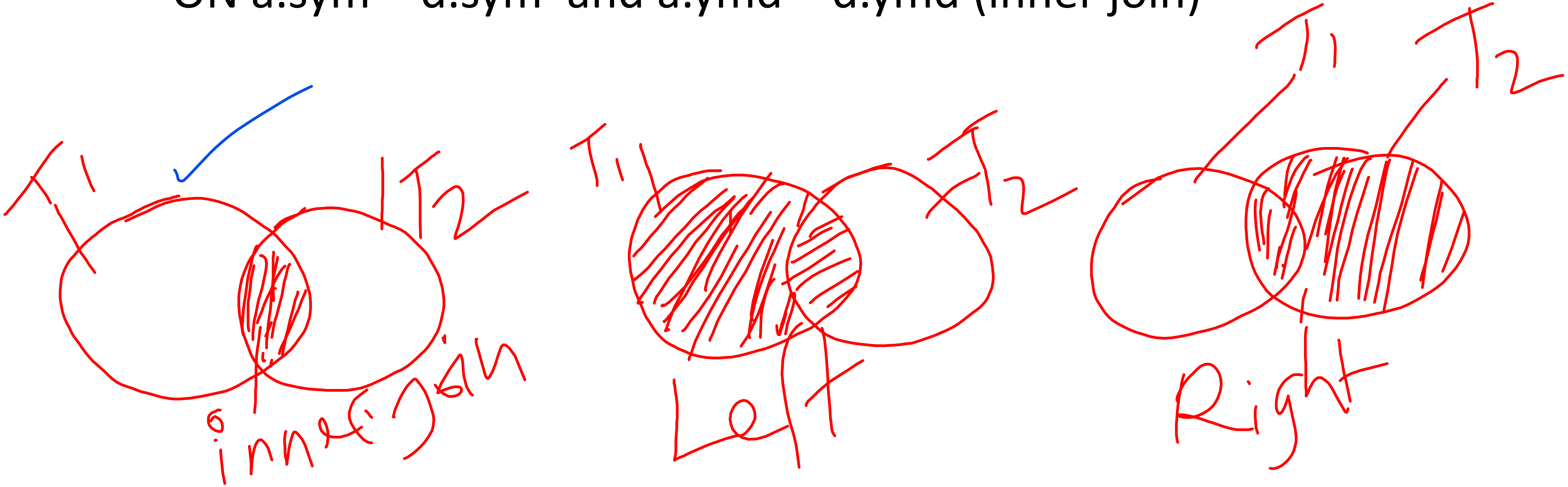
- Create database/ schema/ table/view
- Describe database/table/ view
- Drop database/ table/ view
- Truncate table
- Alter table
- Show database/ tables

Hive DML

- Load files into tables
- Inserting data into hive table from queries

Join

- Select a.sym, a.ymd, a.price_close From stocks a
  Inner Join dividends d
  ON a.sym = d.sym and a.ymd = d.ymd (inner join)



Sqoop

- A command-line interface app for transferring data between relational structured databases and Hadoop
- Sqoop is used to import data from external datastores into HDFS or related Hadoop eco-systems like Hive and Hbase
- Similarly, the other way also possible -- to extract data from Hadoop or its eco-systems and export it to external datastores such as relational databases.

Connection to mysql

- Connect to mysql:

```
mysql --host=192.168.88.137 --user=root --password=bigdata
```

- mySQL comes with mysqldump to extract data from a table and put it in a delimited text file

- Why do we then need tool like sqoop?

Advantages of using Sqoop

- Sqoop → parallelism

→mapreduce/Hadoop framework for data extraction

→ sqoop import– mapreduce map only job is created with multiple mappers

→ each mapper extracts a portion of the content and put them directly into hdfs or even to a hive table

→ for huge database this is a major advantage

→ create sqoop job to import data from database in an incremental fashion

Very first Sqoop command

- sqoop import --connect jdbc:mysql://192.168.88.137/retail_db --table departments --username root --password bigdata --target-dir /user/osboxes/data1
- 4 default mappers
- Sqoop decides what data to send based on the record id.
- We can see the output of the 4 mappers in the hdfs location
- [osboxes@quickstart-bigdata ~]$ sqoop import --connect jdbc:mysql://192.168.88.137/retail_db --table departments --username root --password bigdata → the output folder will be named by table

Override number of mappers ✓

- It is possible to override the number of mappers

- sqoop import --connect jdbc:mysql://192.168.88.137/retail_db --table departments -m 2 --target-dir /user/…

✓

Changing the delimiter from by default space

- sqoop import --connect jdbc:mysql://192.168.88.137/retail_db --table departments --username root --password bigdata -m 1 --target-dir /user/osboxes/stocks_terminated --fields-terminated-by '\t’

  --enclosed-by ‘“’
- Here, it is tab delimited
- And fields are enclosed in double quotation

Selective column and row import

```
sqoop import --connect jdbc:mysql://192.168.88.137/retail_db --table departments --username root --password bigdata --columns "department_id, department_name" --where “department_id > 5" -m 1 --target-dir /user/osboxes/stocks_selective
```

Importing data into Hive or HBase

-
  ```
  sqoop  --connect "jdbc:mysql://localhost/training"
   --username root -P  --table cityByCountry  --target-dir
  /user/where_clause  --where "state = 'Alaska'" --import -hive  -m 1
  ```

Export

- sqoop export --connect jdbc:mysql://localhost/cloudera --username cloudera -P

```
--table exported  --export-dir /user/country_imported/part-m-00000
```

Here exported is the table name

NoSQL – Not only SQL

Job site name

Personal information: name, sin, age, …

Education:…..

Professional info:….

Variety of info/ Sparse data ➔ RDBMS is not suitable

RDBMS → good for fixed/static schema

Need for NoSQL database is practical