hive> set; --> prints all variables from these 4 namespaces:

hive> show databases;

hive> show tables;

hive> show databases like 'f\*';

hive> use default;

hive (default)> use test;

hive> create database if not exists financials;

hive> create database financials

> comment 'holds all financial tables'

> location '/tmp/hive\_db\_dir';

hive> create table records (year string, temperature int, quantity int)

> row format delimited

> fields terminated by '\t';

hive> create table employees (

> name string,

> salary float,

> subordinates array<string>,

> deductions map<string, float>,

> address struct<street:string, city:string, state:string, zip:int>);

hive> create database financials2

> with dbproperties('creator' = 'Ganesh Pillai', 'date' = '2015-02-19');

[cloudera@localhost ~]$ cat scripts/create\_external\_table.hql;

create external table if not exists mydb.stocks(

exchange string,symbol string,ymd string,

price\_open float,

price\_high float,

price\_los float,

price\_close float,

volume int,

price\_adj\_close float)

row format delimited fields terminated by ',' location '/user/cloudera/myhive';

hive (mydb)> create external table stocks2

> like stocks;

hive> create external table mydb.employees3

> like employees;

hive> load data local inpath 'data/sample.txt'

> overwrite into table records;

hive (mydb)> load data local inpath 'data/NYSE\_daily'

> overwrite into table stocks;

Hive> insert overwrite table nyse\_daily\_bak select \* from nyse\_daily;

NOTE: Launges Map-only job

hive> insert overwrite dividends\_bak select \* from dividends;

hive> drop table records;

hive (financials2)> drop database if exists financials2 cascade;

--NOTE: cascade/restrict is optional

hive> quit;

hive> select year, MAX(temperature) from records

> group by year;

hive> describe database financials;

hive> describe database default;

hive (mydb)> describe database mydb;

hive (mydb)> describe employees;

hive (mydb)> describe extended employees;

hive (mydb)> describe formatted employees;

hive (mydb)> describe employees.address;

hive> describe database extended financials2;

OK

financials2 hdfs://localhost.localdomain:8020/user/hive/warehouse

Time taken: 0.124 seconds

hive (financials2)> alter database financials set dbproperties('modified by' = 'GANESH PILLAI

hive (mydb)> create table mydb.employees2

> like employees;

hive (mydb)> create table financials.employees3 like employees;

hive (mydb)> load data local inpath 'data/NYSE\_daily' into table stocks2;

hive> drop database mydb;

FAILED: InvalidOperationException(message:Database mydb is not empty)

hive> drop database mydb cascade;

[cloudera@localhost ~]$ cat scripts/stocks.hql;

create external table if not exists mydb.stocks(exchange string,

symbol string,

ymd string,

price\_open float,

price\_high float,

price\_los float,

price\_close float,

volume int,

price\_adj\_close float)

row format delimited fields terminated by '\t' location '/user/cloudera/myhive';

[cloudera@localhost ~]$ cat scripts/load\_stocks.hql;

load data local inpath 'data/NYSE\_daily'

overwrite into table mydb.stocks;

[cloudera@localhost ~]$ cat scripts/employees\_no\_partition.hql;

create table mydb.employees\_no\_partition(name string,

salary float,

subordinates array<string>,

deductions map<string, float>,

address struct<street:string, city:string, state:string, zip:int>)

row format delimited fields terminated by '\001'

collection items terminated by '\002'

map keys terminated by '\003'

lines terminated by '\n' stored as textfile;

[cloudera@localhost ~]$ cat scripts/load\_employees\_no\_partition.hql;

load data local inpath 'data/employees.txt'

overwrite into table mydb.employees\_no\_partition;

hive (mydb)> select name, subordinates[0] from employees\_no\_partition;

hive (mydb)> select name, address from employees\_no\_partition;

name address

John Doe {"street":"1 Michigan Ave.","city":"Chicago","state"

hive (mydb)> select name, deductions["State Taxes"] from employees\_no\_partition;

name \_c1

John Doe 0.05

hive (mydb)> select name, address.city from employees\_no\_partition;

name city

John Doe Chicago

hive (mydb)> select symbol, `price.\*` from stocks limit 5;

symbol price\_open price\_high price\_los price\_close

CLI 35.39 35.7 34.5

hive (mydb)> select upper(name),salary,deductions["Federal Taxes"],round(salary\*(1-deductions[

\_c0 salary \_c2 \_c3

JOHN DOE 100000.0 0.2 80000.0

hive (mydb)> select count(\*) as count, avg(salary) as salary from employees\_no\_partition;

count salary

7 102857.14285714286

The Numerical functions are listed below in alphabetical order. Use these functions in SQL

ABS( double n )

The ABS function returns the absolute value of a number.

Example: ABS(-100)

ACOS( double n )

Example: ACOS(0.5)

ASIN( double n )

The ASIN function returns the arc sin of value n. This function returns Null if the value n

Example: ASIN(0.5)

BIN( bigint n )

The BIN function returns the number n in the binary format.

Example: BIN(100)

CEIL( double n ), CEILING( double n )

The CEILING or CEILING function returns the smallest integer greater than or equal to the decimal

Example: CEIL(9.5)

CONV( bigint n, int from\_base, int to\_base )

The CONV function converts the given number n from one base to another base.

Example: CONV(100, 10,2)

COS( double n )

hive (mydb)> select count(\*), avg(salary) from employees\_no\_partition;

hive (mydb)> set hive.map.aggr = true;

hive (mydb)> select MAX(column\_name) from tablename;

hive (mydb)> select MIN(column\_name) from tablename; hive (mydb)> select count(distinct symbol) from stocks;

237

hive (mydb)> select count(distinct ymd), count(distinct volume) from stocks;

\_c0 \_c1

explode, json\_tuple, parse\_url\_tuple, stack

NOTE: aggregate : many rows –> one result

table generating funtions: one column –> many rows/columns

OTHER BUILT-IN FUNCTIONS

test in(v1, v2, v3, ...) --> returns true if a match is found

length(s)

reverse(s)

concat(s1,s2,s3,...)

concat(separator,s1,s2,s3,...)

substr(s, start, length)

upper(s)/ucase(s),

trim(s), ltrim(s)

regexp\_replace(s, regex, replacement) --> regexp\_replace(hive, [iv], x) --> hxvx

size(map<k.v>)

siz(array<T>)

cast(<expr> as <type>) --> cast('1' as int)

year, month, day from timestamp

space(n)

repeat(s,n)

split(s, pattern)

instr(str, substr)

in\_file(s, filename)

hive (mydb)> select explode(subordinates) from employees\_no\_partition limit 1;

hive> from (select name, salary from employees\_no\_partition limit 10) e select e.name, e.salary

hive> select name, salary,

> case

> when salary < 50000 then 'low'

> when salary >=50000 and salary <70000 then 'middle'

> when salary >=70000 then 'high'

> else 'very high'

> end

> as salary\_bracket

> from employees\_no\_partition;

hive (mydb)> select \* from employees\_no\_partition;

FORCING HIVE TO AVOID MAP REDUCE:

hive (mydb)> set hive.exec.mode.local.auto = true;

NOTE: No gurantee that Hive won’t invoke map reduce

hive (mydb)> select name from employees\_no\_partition where address.street like '%Main%';

hive (mydb)> select name from employees\_no\_partition where address.street rlike '.\*(Chicago

hive (mydb)> select max(price\_open) from stocks;

hive (mydb)> select symbol,max(price\_open) from stocks;

FAILED: SemanticException [Error 10025]: Line 1:7 Expression not in GROUP BY key 'symbol'

hive (mydb)> select symbol,max(price\_open) from stocks GROUP BY symbol limit 5;

hive (mydb)> select symbol,max(price\_open) as max from stocks GROUP BY symbol having max >

symbol max

CACI 49.79

CAF 39.5

CAH 66.9

CAJ 53.22

CAM 42.47

JOINS

Hive supports:

INNER JOIN

LEFT OUTER JOIN

RIGHT OUTER JOIN

FULL OUTER JOIN

LEFT SEMI JOIN(returns records from the left table if records are found

in right table that satisfy ON predicates)

CARTISIAN PRODUCT JOIN (select \* from T1 join T2)

MAP SIDE JOIN(small table cached in memory)(small table on right side of join)

–>Hive doesn’t support OR between predicates in ON clause (AND is supported)

–>You can join more than 2 tables

ORDER BY, SORT BY, DISTRIBUTED BY WITH SORT BY, CLUSTER BY

ORDER BY –> total ordering

SORT BY –> orders data within each reducer(local ordering)

if hive.mapred.mode=strict –> LIMIT needed with ORDER BY

DISTRIBUTED BY controls how map output is divided among reducers

CLUSTER BY = DISTRIBUTE BY + SORT BY

–NOTE: Comparison:

ITEM ORDERING # REDUCERS OUTPUT

ORDER BY Global 1 reducer(unacceptable for large data sets) 1 sorted le

SORY BY Local to reducer N reducers(with overlapping data) >=N sorted les

DISTRIBUTE BY No sorting N reducers(non-overlapping data)(not sorted) >=N sorted les

CLUSTER BY [result:global] N reducers(non-overlapping data)(sorted) >=N sorted les

hive (mydb)> select s.ymd, s.symbol, s.price\_close from stocks s order by s.ymd asc, s.symbol

hive (mydb)> select s.ymd, s.symbol, s.price\_close from stocks s sort by s.ymd asc, s.symbol

hive (mydb)> select s.ymd, s.symbol, s.price\_close from stocks s distribute by s.symbol sort

hive (mydb)> select s.ymd, s.symbol, s.price\_close from stocks s cluster by s.symbol;

SORT BY: (each reducer output is ordered; but total order is missing)

r1: 0

0

3

9

r2: 0

0

1

2

ORDER BY: (single output; fully ordered)

r: 0

0

0

0

1

2

3

9

DISTRIBUTE BY: (same keys go into same reducer; no guarantee to be clusteed in adjacent positions)

r1: x1

x2

x1

r2: x4

x3

CLUSTER BY: (global ordering across multiple reducers)

r1: x1

x1

x2

r2: x3

x4

hive (mydb)> select name from employees\_no\_partition where cast(salary as int) > 100000;

hive (mydb)> select name from employees\_no\_partition where cast(salary as float) > 100000.0;

hive (mydb)> create external table numbers(number int);

hive (mydb)> load data local inpath 'data/numbers.txt' into numbers;

hive (mydb)> describe numbers;

OK

col\_name data\_type comment

number int

hive (mydb)> select \* from numbers;

number

1

2

3

4

5

6

7

8

10

hive (mydb)> select \* from numbers TABLESAMPLE(BUCKET 2 OUT OF 5 on rand()) s;

number

7

8

10

SAMPLING

–> tablesample(n percent) –> n % of the data size (not n% of rows)

–NOTE: Smallest unit of sampling is a single HDFS block; if table

size > block size, all rows are returned;

–> In percentage sampling, by using ‘set hive.sample.seednumner=<integer>’ you can control the subsets of data

sampled

x1

x2

r2: x3

x4

hive (mydb)> select name from employees\_no\_partition where cast(salary as int) > 100000;

hive (mydb)> select name from employees\_no\_partition where cast(salary as float) > 100000.0;

hive (mydb)> create external table numbers(number int);

hive (mydb)> load data local inpath 'data/numbers.txt' into numbers;

hive (mydb)> describe numbers;

OK

col\_name data\_type comment

number int

hive (mydb)> select \* from numbers;

number

1

2

3

4

5

6

7

8

10

hive (mydb)> select \* from numbers TABLESAMPLE(BUCKET 2 OUT OF 5 on rand()) s;

number

7

8

10

hive (mydb)> select \* from numbers TABLESAMPLE(BUCKET 2 OUT OF 5 on rand()) s;

OK

number

hive (mydb)> select \* from numbers TABLESAMPLE(BUCKET 2 OUT OF 5 on rand()) s;

OK

number

5

hive> select \* from numbers TABLESAMPLE(BUCKET 1 OUT OF 5 on number) s;

OK

1

6

hive> select \* from numbers TABLESAMPLE(BUCKET 1 OUT OF 5 on number) s;

OK

1

6

hive> select \* from numbers TABLESAMPLE(BUCKET 1 OUT OF 5 on number) s;

OK

1

6

hive> select \* from numbers TABLESAMPLE(BUCKET 1 OUT OF 5 on number) s;

OK

5

10

hive> select \* from numbers TABLESAMPLE(BUCKET 1 OUT OF 5 on number) s;

OK

5

10

select \* from numbers tablesample(0.1 percent) s;

OK

1

2

3

4

5

6

7

8

10

Time taken: 11.612 seconds

hive> create table numbers\_bucketed(number int) clustered by (number) into 3 buckets;

OK

Time taken: 0.264 seconds

hive> set hive.enforce.bucketing=true;

hive> insert table numbers\_bucketed select number from numbers;

FAILED: ParseException line 1:0 cannot recognize input near 'insert' 'table' 'numbers\_bucketed'

hive> insert overwrite table numbers\_bucketed select number from numbers;

Total MapReduce CPU Time Spent: 20 seconds 600 msec

OK

Time taken: 61.175 seconds

hive> select \* from numbers\_bucketed;

OK

3

6

1

4

7

10

2

5

8

Time taken: 0.711 second

[cloudera@localhost ~]$ hadoop fs -cat /user/hive/warehouse/mydb.db/numbers\_bucketed/000000\_0;

3

6

[cloudera@localhost ~]$ hadoop fs -cat /user/hive/warehouse/mydb.db/numbers\_bucketed/000001\_0;

1

4

7

10

[cloudera@localhost ~]$ hadoop fs -cat /user/hive/warehouse/mydb.db/numbers\_bucketed/000002\_0;

2

5

8

hive> select \* from numbers\_bucketed tablesample(bucket 1 out of 3 on number) s;

OK

3

6

hive> select \* from numbers\_bucketed tablesample(bucket 2 out of 3 on number) s;

OK

1

4

7

10

hive> select \* from numbers\_bucketed tablesample(bucket 3 out of 3 on number) s;

OK

2

5

8

from (from stocks select stocks.symbol, stocks.price\_open where stocks.volume < 100000

> union all

from stocks select stocks.symbol, stocks.price\_open where stocks.volume <150000)

> unioninput

> insert overwrite directory 'output/union.out' select unioninput.\*;

OK

Time taken: 14.794 seconds

hive> select name, size(subordinates) from employees\_no\_partition;

OK

John Doe 2

Mary Smith 1

Todd Jones 0

Bill King 0

Boss Man 2

Fred Finance 1

Stacy Accountant 0

Time taken: 20.045 seconds

hive> create view managers as select \* from employees\_no\_partition where size(subordinates)

OK

Time taken: 1.185 seconds

hive> show tables;

OK

employees

employees\_no\_partition

managers

numbers

numbers\_bucketed

numbers\_clustered

stocks

test

Time taken: 0.484 seconds

hive> select name from managers;

OK

john Doe

Mary Smith

Boss Man

Fred Finance

Time taken: 16.062 seconds

hive>describe formatted managers;

sample\_output:

OK

# col\_name data\_type comment

name string None

salary float None

subordinates array<string> None

deductions map<string,float> None

address struct<street:string,city:string, None

state:string,zip:int>

# Detailed Table Information

Database: mydb

Owner: cloudera

CreateTime: Tue Feb 24 00:08:24 CST 2015

Table Type: VIRTUAL\_VIEW

hive> create table tblmanagers like managers;

OK

Time taken: 2.956 seconds

hive> create external table exttblmanagers like managers;

OK

Time taken: 0.757 seconds

hive> alter view managers set tblproperties('created\_at'='some\_timestamp');

OK

Time taken: 2.435 seconds

create table mydb.employees\_partitioned(name string,salary float,

subordinates array<string>,deductions map<string, float>,

address struct<street:string, city:string, state:string, zip:int>

partitioned by (country string, state string);

create index employees\_partitioned\_index on

table mydb.employees\_partitioned (country) as 'org.apache.hadoop.hive.ql.index.compact.CompactIndexHandler'

with deferred rebuild idxproperties

('creator'='me', 'created\_at'='some\_time')in table mydb.employees\_index\_tablePARTITIONED BY

(country, name)

comment 'employees\_partitioned

by country and name');

——>TUNING

Use [EXTEDNED] EXPLAIN

Tune LIMIT clause (use set )

Tune JOIN clause(use map-sie join on small datasets)

use Local Mode on small datasets(use set)

Parallel Execution (execute jobs in parallel)(use set)

Strict Mode (use set hive.mapred.mode=strict). Enforces the following:

enforces WHERE clause on partitioned tables

enforces LIMIT clause with ORDER BY

enforces ON clause with catesian product (JOIN)

Tuning number of reducers:

set hive.exec.reducers.bytes.per.reducer=750,000,000

set hive.exec.reducers.max (max reducer count)

reduce.tasks (reducer count)

JVM Reuse (use set mapred.job.reuse.jvm.num.tasks=10)

Use Indexes

Tune dynamic partition:

exec.dynamic.partition.mode=strict (at least one static partition is needed)

exec.max.dynamic.partitions

exec.max.dynamic.partitions.pernode

o Speculative Execcution (Hadoop launches multiple instances of map/reduce jobs):

map.tasks.speculative.execution=true;

reduce.tasks.speculative.execution=true;

mapred.reduce.tasks.speculative.execution=true;

Use single mapreduce job to combine multiple GROUP BY in a query(common group by key is rqd)(use set)

Use Virtual columns (see example below)

hive (test)> !pwd;

hive (test)> /home/cloudera

hive> set hive.cli.print.header=true;

[cloudera@localhost ~]$ cat scripts/create\_table.hql

CREATE TABLE IF NOT EXISTS mydb.employees (

name STRING COMMENT ‘Employee name’,

salary FLOAT COMMENT ‘Employee salary’,

subordinates ARRAY<STRING> COMMENT ‘Names of subordinates’,

deductions MAP<STRING, FLOAT> COMMENT ‘Keys are deductions names, values are percentages’,

address STRUCT<street:STRING, city:STRING, state:STRING, zip:INT> COMMENT ‘Home address’)

COMMENT ‘Description of the table’

LOCATION ‘/user/hive/warehouse/’

TBLPROPERTIES (‘creator’=’me’, ‘created\_at’=’2015-02-19 9:55:00’ );

[cloudera@localhost ~]$ hive -f ‘scripts/create\_table.hql’