Hive Database

Create Database

```
create database if not exists testdatabase;

create database if not exists movie comment "This database holds all tables about movies";

Create database accounting  location '/user/cloudera';
```

Show Database

```
show databases;

show databases like 'm*';
```

Describe Database

```
Describe database movie;

Describe formatted database movie;
```

Drop Database

```
Drop database movie;

Drop database movie cascade;
```

Hive Managed Tables

Create Managed Table

```
CREATE TABLE customers
(custId INT, fName STRING, lName STRING, city STRING)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY '|'
STORED AS TEXTFILE;
```

sqoop import --connect "jdbc:mysql://localhost/retail_db" --username root --password cloudera --table customers --target-dir /user/cloudera/customer-hive --fields-terminated-by '|' –delete-target-dir --columns "customer_id,customer_fname,customer_lname,customer_city"

Load Data From HDFS Location

```
LOAD DATA INPATH '/user/cloudera/customer-hive ' overwrite into table customers;
```

Load Data From Local File System

```
LOAD DATA LOCAL INPATH '/user/cloudera/customer-hive ' overwrite into table ‘customers’;
```

Drop Managed Table

```
Drop table customers;
```

BIG DATA | Tech Talk With Navdeep

Hive External Tables

Create External Table

```
CREATE EXTERNAL TABLE customers
(custId INT, fName STRING, lName STRING, city STRING)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY '|'
STORED AS TEXTFILE
LOCATION '/user/cloudera/customer-hive';
```

```
sqoop import --connect "jdbc:mysql://localhost/retail_db" --username root --password cloudera
--table customers --target-dir /user/cloudera/customer-hive --fields-terminated-by '|' –delete-
target-dir --columns "customer_id,customer_fname,customer_lname,customer_city"
```

Drop External Table

```
DROP TABLE customers;
```

Create Table

```
CREATE TABLE customer_new as SELECT * from customers;
```

Hive Analysis

GroupBy/OrderBy Function

```
Select city, count(*)
from customers
group by city
order by city
limit 100;
```

GroupBy/Having/OrderBy Function

```
Select city, count(*)
from customers
group by city
having count(*)>200
 order by city
limit 5;
```

IF clause

```
Select city,
if (count(*) >50,1,0) as bigCity
from customers
group by city;
```

Min/Max/Sum Function

```
Select min(custid),max(custid),sum(custid),avg(custid) from customers;
```

Different formats/Compressions

Parquet Format

```
Create table customer_parquet
stored as parquet
 location '/user/cloudera/customers-parquet'
 as select * from customers;
```

SET parquet.compression=SNAPPY;

SET hive.exec.compress.output=true;

Parquet Format with Snappy compression

```
Create table customer_parquet_snappy
stored as parquet
location '/user/cloudera/customer-parquet-snappy'
 as select * from customers;
```

sqoop import --connect "jdbc:mysql://localhost/retail_db" --username root --password cloudera --table customers --target-dir /user/cloudera/customer-avro --fields-terminated-by '|' --columns "customer_id,customer_fname,customer_lname,customer_city" --as-avrodatafile

avro-tools getschema hdfs://localhost/user/cloudera/customer-avro/part-m-00000.avro > customer.avsc

hdfs dfs -mkdir /user/cloudera/customer-avro-schema

hdfs dfs -put customer.avsc /user/cloudera/customer-avro-schema

Avro Format

```
CREATE EXTERNAL TABLE customers_avro

(custId INT, fName STRING, lName STRING, city STRING)

ROW FORMAT DELIMITED

FIELDS TERMINATED BY '|'

STORED AS AVRO

location '/user/cloudera/customer-avro'

TBLPROPERTIES ('avro.schema.url'='/user/cloudera/customer-avro-schema/customer.avsc');
```

Fixed File format using Regular Expressions

```
Create External Table employee_fixed
(empId int,
name STRING,
age int)
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.RegexSerDe'
with SERDEPROPERTIES("input.regex" = "(.{4})(.{10})(.{2})" )
location '/user/cloudera/employee-fixed';
```

Avro Format with Snappy Compressions

```
CREATE TABLE customers_avro_snappy
STORED AS AVRO
location '/user/cloudera/customer-avro-snappy'
TBLPROPERTIES ('avro.schema.url'='/user/cloudera/customer-avro-schema/customer.avsc', "avro.output.codec"="snappy")
as Select * from customers;
```

Hive String Functions

Concat Function

```
select
concat(fname,'+',lname,':',city)
from customers;
```

Concat_ws Function

```
Select
concat_ws(' ',fname,lname,'is from',city)
from customers;
```

Length Function

```
select
concat(fname,' ',lname)
from customers
where length(concat(fname,lname))>8;
```

Lower/Upper/Reverse Function

```
Select
 UPPER(fname),LOWER(lname),REVERSE(city)
from customers;
```

LPAD/RPAD Function

```
Select
 LPAD(fname,6,'#'),RPAD(lname,6,'*')
from customers;
```

Split Function

```
select
split(fname,'a')
from customers;
```

SUBSTR Function

```
select
SUBSTR(fname,3)
from customers;
```

Regexp_extract Function

```
Select
 regexp_replace(city,'Las','Allas')
from customers
where city like '%Las%';
```

Hive Date Functions

Month,Year,Day Function

```
select
year('1970-01-01'),
month('1970-01-11'),
day('1970-01-11')
from customers limit 1
```

Hour/Minute/Second Function

```
Select
hour('2009-07-30 12:58:59'),
minute('2009-07-30 12:58:59'),
second('2009-07-30 12:58:59')
from customers limit 1;
```

DateDiff Function

```
select
datediff('2019-03-01', '2019-02-27')
from customers limit 1;
```

Date_Add/Date_Sub Function

```
select
date_add('2019-01-26', 2),
date_sub('2019-12-31', 1)
 from customers limit 1;
```

Add_months/months_between Function

```
select
add_months('2019-01-26', 2),
months_between('2019-04-01', '2019-01-01')
 from customers limit 1;
```

Current_date/current_timestamp Function

```
select
current_date,
current_timestamp
from customers limit 1;
```

Unix_Timestamp Function

```
select
unix_timestamp()
from customers limit 1;
```

Date to Unix_Timestamp conversion

```
select unix_timestamp('2019-03-20 11:30:01') from customers limit 1;
select unix_timestamp('2019/03/20','yyyy/MM/dd') from customers limit 1;
```

BIG DATA | Tech Talk With Navdeep

Hive Partioning

sqoop import --connect "jdbc:mysql://localhost/retail_db" --username root --password cloudera --table orders --target-dir "/user/cloudera/orders"

Order Table

```
CREATE EXTERNAL TABLE orders
(ordid INT, date STRING, custid INT,status STRING)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
Location '/user/cloudera/orders';
```

Partitioned based on status

```
CREATE EXTERNAL TABLE orders_partitioned
(ordid INT, date STRING, custid INT)
PARTITIONED BY (status STRING)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ',' ;
```

```
set hive.exec.dynamic.partition=true;
set hive.exec.dynamic.partition.mode=nonstrict;
```

Populate partitions

```
INSERT OVERWRITE TABLE  orders_partitioned
PARTITION (status)
SELECT ordid, date, custid,status from orders;
```

Show partitions

```
SHOW PARTITIONS orders_partitioned;
```

```
hive> SHOW PARTITIONS orders_partitioned;
OK
status=CANCELED
status=CLOSED
status=COMPLETE
status=ON_HOLD
status=PAYMENT_REVIEW
status=PENDING
status=PENDING_PAYMENT
status=PROCESSING
status=SUSPECTED_FRAUD
Time taken: 0.32 seconds, Fetched: 9 row(s)
```

Hive Bucketing

Bucketed on custID

```
CREATE EXTERNAL TABLE orders_bucketed
(ordid INT, date STRING, custid INT,status STRING)
CLUSTERED BY (custid) into 10 BUCKETS
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ',' ;
```

```
set hive.enforce.bucketing = true;
```

Populate buckets

```
INSERT OVERWRITE TABLE  orders_bucketed
SELECT ordid, date, custid,status from orders;
```

Partitioned based on status/Bucketed on date

```
CREATE EXTERNAL TABLE orders_partitioned_bucketed

(ordid INT, date STRING, custid INT)

PARTITIONED BY (status STRING)

CLUSTERED BY (custid) into 10 BUCKETS

ROW FORMAT DELIMITED

FIELDS TERMINATED BY ',' ;
```

Populate partitions

```
INSERT OVERWRITE TABLE  orders_partitioned_bucketed
PARTITION (status)
SELECT ordid, date, custid,status from orders;
```