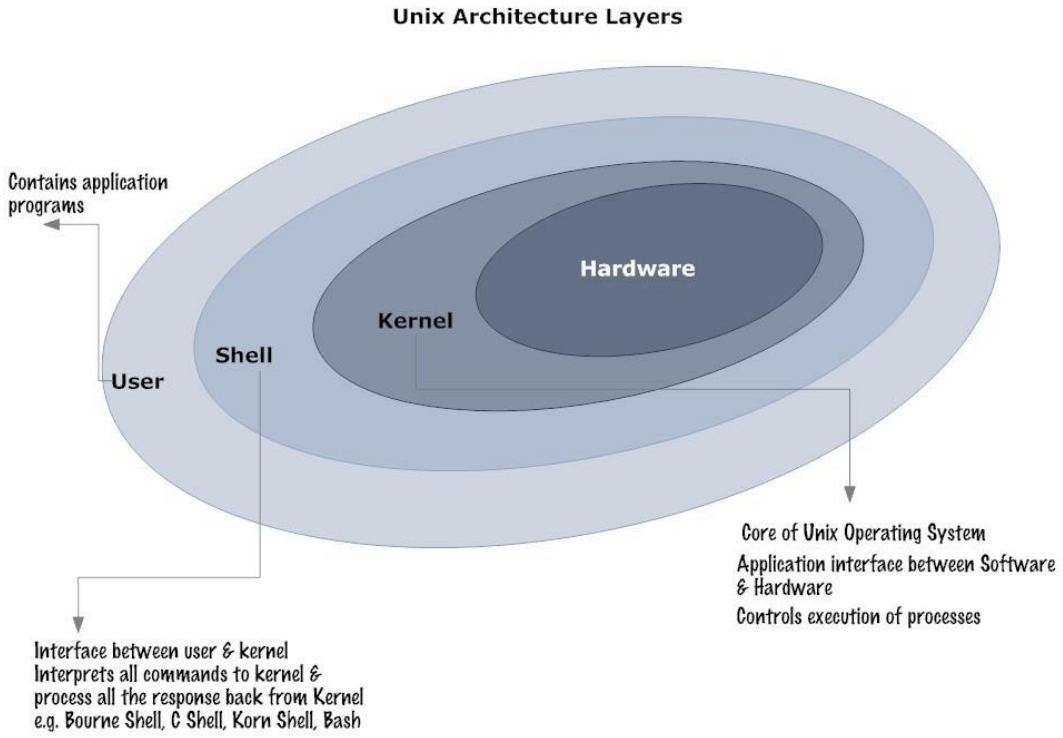
Web: Inceptez.com Mail: info@inceptez.com Call: 7871299810, 7871299817

LINUX

UNIX is a computer Operating System which is capable of handling activities from multiple users at the same time performing multitasking of programs. UNIX was originated around in 1969 at AT&T Bell Labs by Ken Thompson and Dennis Ritchie.

Linux Architecture



Architecture Components

- **Kernel:** The kernel is the heart of the operating system. It interacts with hardware and most of the tasks like memory management, task scheduling and file management.

- **Shell:** The shell is the utility that processes your requests. When you type in a command at your terminal, the shell interprets the command and calls the program that you want. C Shell, Bourne Shell and Korn Shell are most famous shells which are available with most of the Unix variants.

- **Commands and Utilities:** There are various command and utilities which you would use in your day to day activities. **cp, mv, cat** and **grep** etc. are few examples of commands and utilities. There are over 250 standard commands plus numerous others provided through 3rd party software. All the commands come along with various optional options.

- **Files and Directories:** All data in UNIX is organized into files. All files are organized into directories. These directories are organized into a tree-like structure called the filesystem.

Development Commands

Files and Directory management

Files

Different files are available such as flat files, compressed files, hidden files and system files.

Listing Files

Short list

ls

Long list

ls -l

Long list all files and directories sorting by modification time in descending order

ls -lart

Details about all the listed columns −

- First Column: represents file type and permission given on the file. Below is the description of all type of files.
- Second Column: represents the number of dirs/links in the directory.
- Third Column: represents owner of the file. This is the Unix user who created this file.
- Fourth Column: represents group of the owner. Every Unix user would have an associated group.
- Fifth Column: represents file size in bytes.
- Sixth Column: represents date and time when this file was created or modified last time.
- Seventh Column: represents file or directory name.

Listing hidden files

ls -a

File handling

Create/Edit file

vi file1

Add some content in the above file after typing '**i**'.

Delete a line : esc+dd

Delete a word : esc+dw

Delete a character : esc+x

Search for a word: esc /wordtofind

Goto a line number: esc :linenumber

Undo: esc+u

Redo: ctrl+R

To save type **Shift+: wq**

To quit without saving **Shift+: q!**

Create/Replace a file with few content

echo 'welcome to unix' > filename1

Append content to a file

echo "this is the second line" >> filename1

Create Empty file

touch filename2

> filename3

Display content of a file

Display whole content

cat filename

Display incremental content

more filename

Display first 10 lines

head filename

Display last 10 lines

tail filename

Counting number of lines in a file

wc -l filename

File operation (copy, move, rename, delete)

cp filename file2

mv file2 file3

rm file3

Directory Commands

You can go in your home directory anytime using the following command –

cd ~

Here ~ indicates home directory. If you want to go in any other user's home directory then use the following command –

cd ~hduser

To go in your last directory you can use following command –

cd -

To go to the parent directory

cd ..

cd ../..

Absolute path

Fully qualified path start with '/' provided from root till the child.

mkdir -p /home/hduser/tmp/mylogs

cd /home/hduser/tmp/mylogs

Relative path

Access the rest of the child path from the parent path.

cd ~

cd tmp/mylogs

cd -

Create Dir

mkdir dirname

Create directory structure from parent directory, here all three dir1,2,3 will be created using option 'p'.

mkdir -p /home/hduser/dir1/dir2/dir3

Change Dir

cd dirname

cd ..

Move dir

mv dirname dirname1

Remove Dir

rmdir ~/dirname1

Disk size commands

Disk usage

du -k /tmp

Grep

grep 2 filename

Execute a shell script

vi scriptname.bsh

```
echo "script is executing"
echo "number of params are $#"
echo "script name is $0"
echo "param one is $1"
echo "param one is $2"
cp -p $1/$2 /tmp/
head -10 $1/$2 >> $1/filename10
```

Executing the script with the parameters

bash scriptname.bsh /home/hduser filename

Compression

gzip filename

gunzip filename.gz

HDFS Commands

Open a terminal window to the current working directory.

```
cd /home/hduser
```

1. Print the Hadoop version

```
hadoop version
```

2. Report the amount of space used and # available on currently mounted filesystem

```
hadoop fs -df hdfs:/
```

3. Count the number of directories, files and bytes under # the paths that match the specified file pattern #

```
hadoop fs -count hdfs:/
```

4. Count the number of directories, files and bytes under # the paths that match the specified file pattern #

```
hadoop fs -mkdir /user/hduser/hadoop
```

```
hadoop fs -mkdir -p /user/hduser/hadoop/dir1/dir2
```

5. Create a sample file in linux and place it into hadoop directory

```
echo "sampledata" > sample.txt
```

hadoop fs -put ~/sample.txt /user/hduser/hadoop

hadoop fs -copyFromLocal -f ~/sample.txt /user/hduser/hadoop

6. List the contents of this new directory in HDFS.

hadoop fs -ls /user/hduser/hadoop

7. Copy a directory from local to hadoop.

hadoop fs -put /home/hduser/mrdata /user/hduser/hadoop

8. Remove a HDFS file

hadoop fs -copyFromLocal test1.txt hadoop/

hadoop fs -rm hadoop/test1.txt

10. Remove the entire directory and all of its contents in hadoop.

hadoop fs -mkdir hadoop/test

hadoop fs -put ~/sample.txt hadoop/test

hadoop fs -rm -r hadoop/test

11. Copy the file from hadoop to local.

hadoop fs -copyToLocal /user/hduser/test.txt /tmp

12. Remove all files from hadoop directory ending with .txt

hadoop fs -rm hadoop/*.txt

13. cp is used to copy files between directories present in HDFS

hadoop fs -cp /user/hduser/test.txt /user/hduser/test2.txt

14. Get command to copy the file from hadoop to local.

hadoop fs -get test2.txt /home/hduser/test3.txt

15. Display last few lines in hadoop

hadoop fs -put filename

hadoop fs -tail filename

16. HDFS zero byte file

```
hadoop fs -touchz hadoop/test4.txt
```

```
hadoop fs -ls hadoop/test4.txt
```

17. View the content of copied file.

```
hadoop fs -cat /user/hduser/testing/test.txt
```

18. Move file from local to hdfs

```
hadoop fs -moveFromLocal ~/test.txt /user/hduser/test.txt
```

19. Append file from local to hdfs.

```
cd ~
```

```
echo somedata > test1.txt
```

```
hadoop fs -appendToFile test1.txt /user/hduser/testing/test.txt
```

SQOOP WORKOUTS

MYSQL (Preparation of Source) :

Start the MYSQL Service :

```
sudo service mysqld start

mysql -u root -p

Password: root
```

Select the custdb database:

```
create database custdb;

use custdb;

CREATE TABLE customer (custid INT,firstname VARCHAR(20),lastname VARCHAR(20),city varchar(50),age int,createdt date,transactamt int );

insert into customer values(1,'Arun','Kumar','chennai',33,'2017-09-20',100000);

insert into customer values(2,'srini','vasan','chennai',33,'2017-09-21',10000);

insert into customer values(3,'vasu','devan','banglore',39,'2017-09-23',90000);

insert into customer values(4,'mohamed','imran','hyderabad',33,'2017-09-24',1000);

insert into customer values(5,'arun','basker','chennai',23,'2017-09-20',200000);

insert into customer values(6,'ramesh','babu','manglore',39,'2017-09-21',100000);

create table customer_bkp as select * from customer;

create table customer_bkp1 as select * from customer;

select * from customer;
```

SQOOP WORKOUTS (Open a separate linux terminal)

To List Databases which are in MySql

```
sqoop list-databases --connect jdbc:mysql://localhost --username root --password root;
```

To List Tables from custdb database

```
sqoop list-tables --connect jdbc:mysql://localhost/custdb --username root --password root;
```

Import Table from SQL to HDFS with 1 mapper:

```
sqoop import --connect jdbc:mysql://localhost/custdb --username root --password root -table customer -m 1 --delete-target-dir ;
```

Check whether the below import works?

```
sqoop import --connect jdbc:mysql://localhost/custdb --username root --password root -table customer -m 2;
```

```
sqoop import --connect jdbc:mysql://localhost/custdb --username root -P -table customer -m 3 \
--split-by custid --target-dir sqoop_import --delete-target-dir --direct;
```

```
sqoop import --connect jdbc:mysql://localhost/custdb --username root --password root -table customer -m 3 \
--split-by city --append --fetch-size 100;
```

Fields terminated by & Lines terminated by

```
sqoop import --connect jdbc:mysql://localhost/custdb --username root --password root -table customer -m 1 --target-dir imp_del --fields-terminated-by '~' --lines-terminated-by '\n' --delete-target-dir;
```

Controlling Import:

Using Where condition:

```
sqoop import --connect jdbc:mysql://localhost/custdb --username root --password root --table customer -m 1 --where "city ='banglore' or  age>33" --target-dir filtered  --delete-target-dir;
```

Using free form query:

```
sqoop import --connect jdbc:mysql://localhost/custdb --username root --password root --query " select custid,age,transactamt from customer where (city ='banglore' or  age>33) and \$CONDITIONS  " --target-dir filtered  --delete-target-dir -m 2 --split-by custid;
```

Export from HDFS to SQL:

Create table in MYSQL before running this command

```
CREATE TABLE customer_hdfs (custid INT,firstname VARCHAR(20),lastname VARCHAR(20),city varchar(50),age
```

int,createdt date,transactamt int);

sqoop export --connect jdbc:mysql://localhost/custdb --username root --password root --table customer_hdfs \
--export-dir savedjob1

Sqoop Additional commands and Use Cases

Common Usecase 1:

Import All tables from a DB :

sqoop import-all-tables --connect jdbc:mysql://localhost/custdb --username root --password root --warehouse-dir '/user/hduser/sqoop/testtables' -m 1

Import All tables other than excluded tables from a DB :

sqoop import-all-tables --connect jdbc:mysql://localhost/custdb --username root --password root --warehouse-dir '/user/hduser/sqoop/testtables' --exclude-tables customer_bkp1 -m 1

Sqoop evaluation:

sqoop eval --connect jdbc:mysql://localhost/custdb --username root --password root --query "select * from customer "

Batch export:

sqoop export -Dsqoop.export.statements.per.transaction=10 --connect jdbc:mysql://localhost/custdb --username root --password root --table customer1 --export-dir savedjob1 --batch

Usecase 2 :

Create the below Tables MYSQL for Import:

use custdb;

CREATE TABLE customers (custid INT,firstname VARCHAR(20),lastname VARCHAR(20),city varchar(50),age int,createdt date);

CREATE TABLE customer_details (custid INT,firstname VARCHAR(20),fulladdress VARCHAR(200),category varchar(50),transactiondt date,transactamt int,createdt date);

ALTER TABLE customers ADD PRIMARY KEY(custid);

insert into customers values(1,'karthik','vijay','chennai',5,'2018-02-01');

insert into customers values(2,'arun','kumar','chennai',25,'2018-01-30');

```
insert into customers values(3,'vishwa','ajit','hyderabad',null,'2018-02-03');

insert into customers values(4,'bala','palani','bangalore',30,'2018-02-02');

insert into customer_details values(1,'karthik','3/2, jeyaram street, chrompet,Chennai','household','2018-02-01',4000,'2018-02-01');

insert into customer_details values(1,'karthik','3/2, jeyaram street, chrompet,Chennai 44','Automobile','2018-02-02',6000,'2018-02-02');

insert into customer_details values(1,'karthik','3/2, jeyaram street, chrompet,Chennai 44','Foods','2018-02-02',3000,'2018-02-02');

insert into customer_details values(1,'karthik','3/2, jeyaram street, chrompet,Chennai 44',null,'2018-02-02',1000,'2018-02-03');

insert into customer_details values(2,'arun','11, palayam blvd, broadway,Chennai 01','tools','2018-02-02',11000,'2018-02-03');

insert into customer_details values(2,'arun','11, palayam blvd, broadway,Chennai 01','electronics','2018-02-02',15000,'2018-02-04');

insert into customer_details values(3,'vishwa','1A, Elango nagar, Vadapalani,Chennai 33','clothes','2018-02-02',15000,'2018-02-04');


Select a.custid master_custid,a.firstname,b.custid detail_custid,a.createdt,a.age,category,transactamt
from customers a join customer_details b
on a.custid=b.custid;
```

Tables for Export:

```
CREATE TABLE customer_stage (custid INT,fullname VARCHAR(40), city varchar(50),age int,createdt date );

CREATE TABLE customer_exp (custid INT,fullname VARCHAR(40),city varchar(50),age int,createdt date );

ALTER TABLE customer_exp ADD PRIMARY KEY(custid);
```

Import Approach:

1. Import all columns of customer and customer_details data by joining custid between the 2 tables. customer_details can have columns as given.

2. Both custid should be named as master_custid and detail_custid from customer and customer_details respectively.

3. Use column boundary queries using customer.custid column, split using custid

4. Insert null values in category column and age columns import as NA and age a 0 respectively in the hdfs.

5. Store the output in cust_details hdfs directory.

6. Compress the imported data.

7. Use direct mode to transfer the entire content.

8. Define number of mappers as 3.

9. Use fetch size 100.

10. Once the sqoop is executed view the content in hdfs using the command:

hadoop fs -text cust_details/part-m-0000*.gz

Import Solution Command:

sqoop import --connect jdbc:mysql://localhost/custdb --username root -P --boundary-query "select min(custid), max(custid) from customers" --query 'Select a.custid master_custid,a.firstname,a.age,a.city,b.custid detail_custid,a.createdt,b.fulladdress,category,transactiondt,transactamt from customers a join customer_details b on a.custid=b.custid WHERE $CONDITIONS' --split-by a.custid --target-dir cust_details --null-non-string '0' --null-string 'NA' --compress --direct --num-mappers 3 --fetch-size 100 --delete-target-dir;

Export Approach:

1. Import only the subset of columns from the customer table to the HDFS (custid, concatenation of firstname and lastname,age).

2. Export only subset of the above imported columns to the customer_exp table specifying only these 3 columns.

3. Use batch mode for fast export with the sqoop.export.records.per.statement=5.

4. Use staging table to provide consistent load with the clear staging table option to clean the table before each load.

Export Solution Command:

sqoop import --connect jdbc:mysql://localhost/custdb --username root --password root --query " select custid,concat(firstname,' ',lastname),age from customers where \$CONDITIONS " --target-dir cust_exp --delete-target-dir -m 1;

sqoop export -Dsqoop.export.records.per.statement=5 --connect jdbc:mysql://localhost/custdb --username root --password root --table customer_exp --export-dir cust_exp --batch --staging-table customer_stage --clear-staging-table --columns custid,fullname,age

HIVE Workouts

Login to hive cli

```
cd /home/hduser

hive
```

Create Database

```
create database retail;
```

Select Database

```
use retail;
set hive.cli.print.current.db=true;
```

Create table for storing transactional records

Creating managed tables:

```
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 ','
lines terminated by '\n'
stored as textfile;
```

Load the data into the table [From Linux client]

```
LOAD DATA LOCAL INPATH '/home/hduser/hive/data/txns' INTO TABLE txnrecords;
LOAD DATA LOCAL INPATH '/home/hduser/hive/data/txns' OVERWRITE INTO TABLE
txnrecords;
```

Load the data into the table [From HDFS]

```
cd /home/hduser/hive/data/

hadoop fs -copyFromLocal txns txns1

LOAD DATA INPATH '/user/hduser/txns1' OVERWRITE INTO TABLE txnrecords;
```

Creating External tables:

```
create external table externaltxnrecords (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
```

```
location '/user/hduser/hiveexternaldata';
```

Describing metadata or schema of the table

```
describe formatted txnrecords;
describe formatted externaltxnrecords;
```

Export hive data to a local file:

```
insert overwrite local directory '/home/hduser/exptxnrecords' row format delimited fields terminated by ',' select txnno,txndate,custno,amount, product,city,state,spendby from txnrecords where category='Games';
```

Export hive data to a HDFS file:

```
insert overwrite directory '/user/hduser/exptxnrecords'  select concat(txnno, '|',txndate, '|',custno, '|',amount, '|',product, '|',city, '|',state, '|',spendby) from txnrecords where category='Games';
```

Hive Sqoop Integeration

==========

Create a hive table using sqoop as like mysql table

```
sqoop create-hive-table --connect jdbc:mysql://localhost/test --username hiveuser -P --table customer –hive-table customer;
```

Import data from sqoop directly to hive customer table

```
hadoop fs -rmr /user/hduser/customer

sqoop import --connect jdbc:mysql://localhost/test --username hiveuser -P --table customer –hive-import --hive-table default.customer -m 1
```

Use cases:

Customer Categorization: Find sales based on age group

```
create table customer(custno string, firstname string, lastname string, age int,profession string) row format
delimited fields terminated by ',';

load data local inpath '/home/hduser/hive/data/custs' into table customer;
```

Using a simple join on 2 tables:
==========

```
create table cust_trxn (custno int,firstname string,age int,profession string,amount double,product string)
row format delimited fields terminated by ',';

insert into table cust_trxn
select a.custno,a.firstname,a.age,a.profession,b.amount,b.product from customer a JOIN txnrecords b ON
a.custno = b.custno;

select * from cust_trxn limit 10;
```

Using CASE statement
==========

```
create table cust_trxn_case (custno int,firstname string,age int,profession string,amount double,product
string, level string)
row format delimited fields terminated by ',';

insert overwrite table cust_trxn_case
select * , case
when age<30 then 'low'
when age>=30 and age < 50 then 'middle' when age>=50 then 'old'
else 'others' end
from cust_trxn;

select * from cust_trxn_case limit 100;
```

Using Aggregate function
==========

```
create table cust_trxn_aggr (level string, amount double) row format delimited
fields terminated by ',';

insert overwrite table cust_trxn_aggr
select level,sum(amount) from cust_trxn_case group by level;
```

Banking xml data for serde:

```
<record customer_id="0000-JTALA">
<income>200000</income>
<demographics>
<gender>F</gender>
<agecat>1</agecat>
<edcat>1</edcat>
```

<jobcat>2</jobcat>
<empcat>2</empcat>
<retire>0</retire>
<jobsat>1</jobsat>
<marital>1</marital>
<spousedcat>1</spousedcat>
<residecat>4</residecat>
<homeown>0</homeown>
<hometype>2</hometype>
<addresscat>2</addresscat>
</demographics>
<financial>
<income>18</income>
<creddebt>1.003392</creddebt>
<othdebt>2.740608</othdebt>
<default>0</default>
</financial>
</record>

```
<record customer_id="0000-KDELL">
<income>10000</income>
<demographics>
<gender>M</gender>
<agecat>2</agecat>
<edcat>1</edcat>
<jobcat>2</jobcat>
<empcat>3</empcat>
<retire>1</retire>
<jobsat>1</jobsat>
<marital>1</marital>
<spousedcat>1</spousedcat>
<residecat>4</residecat>
<homeown>0</homeown>
<hometype>3</hometype>
<addresscat>2</addresscat>
</demographics>
<financial>
<income>20</income>
<creddebt>1.002292</creddebt>
<othdebt>2.113208</othdebt>
<default>0</default>
</financial>
</record>
```

Copy the serde jars and other dependent jars to hive/lib

```
cp -p /home/hduser/install/hivexmlserde-1.0.5.3.jar /usr/local/hive/lib/
cp -p /home/hduser/install/mysql-connector-java.jar /usr/local/hive/lib/
cp -p /home/hduser/install/hive-exec-0.14.0.jar /usr/local/hive/lib/
```

create below table

```
CREATE TABLE xml_bank(customer_id STRING, income BIGINT, demographics map<string,string>, financial
map<string,string>)
ROW FORMAT SERDE 'com.ibm.spss.hive.serde2.xml.XmlSerDe'
WITH SERDEPROPERTIES (
"column.xpath.customer_id"="/record/@customer_id",
"column.xpath.income"="/record/income/text()",
"column.xpath.demographics"="/record/demographics/*",
"column.xpath.financial"="/record/financial/*"
)
STORED AS
INPUTFORMAT 'com.ibm.spss.hive.serde2.xml.XmlInputFormat'
OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.IgnoreKeyTextOutputFormat'
location '/user/hduser/xmlserdebank/'
TBLPROPERTIES (
"xmlinput.start"="<record customer",
"xmlinput.end"="</record>"
);
```

Copy file to hdfs

```
hadoop fs -mkdir /user/hduser/xmlserdebank/

dfs -put /home/hduser/hive/data/bankserde.xml /user/hduser/xmlserdebank/;


select * from xml_bank;
```