|  |  |
| --- | --- |
| **EX.NO : 1** | INTRODUCTION TO CLOUD COMPUTING TOOLS |
| **DATE:** |

**Aim**

To study the various cloud computing tools

**Introduction to cloud computing**

Cloud computing is an interconnection of large number of computer which can be normal PC’s and Servers and provide different services the computing and storage. It’s also means that instead of all the computer hardware and software you're using sitting on your desktop, or somewhere inside your company's network, it's provided for you as a service by another company and accessed over the Internet, usually in a completely seamless way. Exactly where the hardware and software is located and how it all works doesn't matter to you, the user—it's just somewhere up in the nebulous "cloud" that the Internet represents. Cloud computing is a buzzword that means different things to different people. For some, it's just another way of describing IT (information technology) "outsourcing"; others use it to mean any computing service provided over the Internet or a similar network; and some define it as any bought-in computer service you use that sits outside your firewall.

**Types of cloud computing**

Three different kinds of cloud computing, where different services are being provided for you. Note that there's a certain amount of vagueness about how these things are defined and some overlap between them.

• **Infrastructure as a Service (IaaS)** means you're buying access to raw computing hardware over the Net, such as servers or storage. Since you buy what y ou need and pay-as-you-go, this is often referred to as utility computing. Ordinary web hosting is a simple example of IaaS: you pay a monthly subscription or a per-megabyte/gigabyte fee to have a hosting company serve up files for your website from their servers.

• **Software as a Service (SaaS)** means you use a complete application running on someone else's system. Web-based email and Google Documents are perhaps the best-known examples. Zoho is another well known SaaS provider offering a variety of office applications online.

**• Platform as a Service (PaaS)** means you develop applications using Web-based tools so they run on systems software and hardware provided by another company. So, for example, you might develop your own ecommerce website but have the whole thing, including the shopping cart, checkout, and payment mechanism running on a merchant's server. App Cloud (from salesforce.com) and the Google App Engine are examples of PaaS.

**Advantages and disadvantages of cloud computing**

**Advantages**

The pros of cloud computing are obvious and compelling. If your business is selling books or repairing shoes, why get involved in the nitty gritty of buying and maintaining a complex computer system? If you run an insurance office, do you really want your sales agents wasting time running anti-virus software, upgrading word-processors, or worrying about hard-drive crashes? Do you really want them cluttering your expensive computers with their personal emails, illegally shared MP3 files, and naughty YouTube videos—when you could leave that responsibility to someone else? Cloud computing allows you to buy in only the services you want, when you want them, cutting the upfront capital costs of computers and peripherals. You avoid equipment going out of date and other familiar IT problems like ensuring system security and reliability. You can add extra services (or take them away) at a moment's notice as your business needs change. It's really quick and easy to add new applications or services to your business without waiting weeks or months for the new computer (and its software) to arrive.

**Disadvantage**

Instant convenience comes at a price. Instead of purchasing computers and software, cloud computing means you buy services, so one-off, upfront capital costs become ongoing operating costs instead. That might work out much more expensive in the long-term. If you're using software as a service (for example, writing a report using an online word processor or sending emails through webmail), you need a reliable, high-speed, broadband Internet connection functioning the whole time you're working. That's something we take for granted in countries such as the United States, but it's much more of an issue in developing countries or rural areas where broadband is unavailable.

**Cloud Computing Tools**

**Microsoft Azure**

Windows Azure, which was later renamed as Microsoft Azure in 2014, is a cloud computing platform, designed by Microsoft to successfully build, deploy, and manage applications and services through a global network of datacenters. Azure platform is developed in such a way that developers need to concentrate on only the development part and need not worry about other technical stuff outside their domain. Thus most of the administrative work is done by Azure itself

**Advantages**

High Efficiency

Security

Cost efficient

Scalability

**Open stack**

OpenStack is a platform for open-source cloud computing. This is deployed as a service solution which consists of integrated project. This helps to control throughout a data centre with a help of a web based dashboard or command-line tools. OpenStack is a 100% python based cloud environment with pre-designed and existing python libraries. Use of MySQL is also integrated in this. OpenStack is a great tool to manage an entire data center environment.

**Advantages**

Rapid Innovation

Scalability

Cost efficient

Resource utilization

**Amazon EC2**

Amazon EC2 instance is nothing but a virtual server in Amazon Web services terminology. It stands for Elastic Compute Cloud. It is a web service where an AWS subscriber can request and provision a compute server in AWS cloud.

An on-demand EC2 instance is an offering from AWS where the subscriber/user can rent the virtual server per hour and use it to deploy his/her own applications.

The instance will be charged per hour with different rates based on the type of the instance chosen. AWS provides multiple instance types for the respective business needs of the user.

**Advantages**

  Highly scalable

Quick

Reliable data storage infrastructure

**Cloud Sim**

Cloud Sim is a software framework that supports several core functionality of cloud like job/task queue, processing of events, creation of cloud entities, communication between entities

**Core features of Cloud Sim:**

Support of modeling and simulation of large scale computing environment.

A self contained platform for modeling clouds, service brokers, provisioning and allocation policies.

Support for simulation of network connections among the simulated system elements.

**Google App Engine**

Google App Engine (often referred to as GAE or simply App Engine) is a web framework and cloud computing platform for developing and hosting web applications in Google-managed data centers. Applications are sandboxed and run across multiple servers. App Engine offers automatic scaling for web applications—as the number of requests increases for an application, App Engine automatically allocates more resources for the web application to handle the additional demand.

**Features**

Data storage, retrieval, and search include features such as HRD migration tool, Google Cloud SQL, logs, datastore, dedicated Memcache,

blob store, Mem cache and search.

Communications include features such as XMPP. channel, URL fetch, mail, and Google Cloud Endpoints.

****

**Result**

Thus the various cloud computing tools are studied successfully

|  |  |
| --- | --- |
| **EX.NO : 2** | FIND PROCEDURE TO RUN THE VIRTUAL MACHINE OF DIFFERENT CONFIGURATION. CHECK HOW MANY VIRTUAL MACHINE CAN BE UTILIZED AT PARTICULAR TIME |
| **DATE:** |

**Aim**

To find procedure to run the virtual machine of different configuration and check how many virtual machine can be utilized at particular time

**Procedure**

1. Download and install oracle virtual box

A screenshot of a cell phone

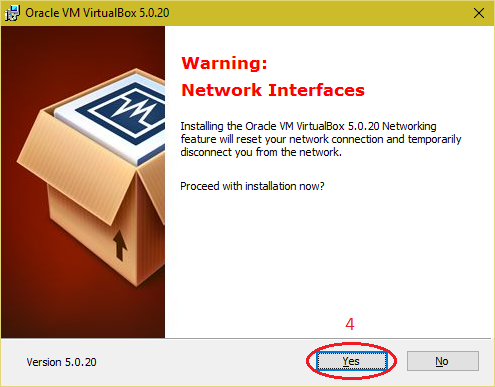
Description generated with very high confidence

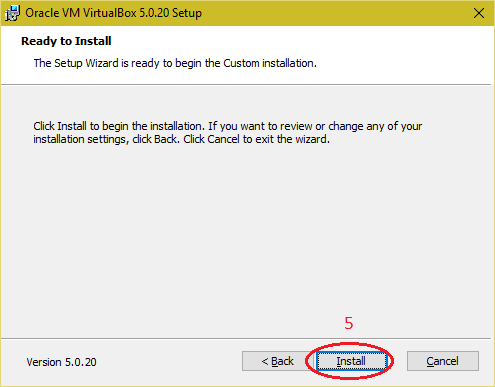
A screenshot of a cell phone

Description generated with very high confidence

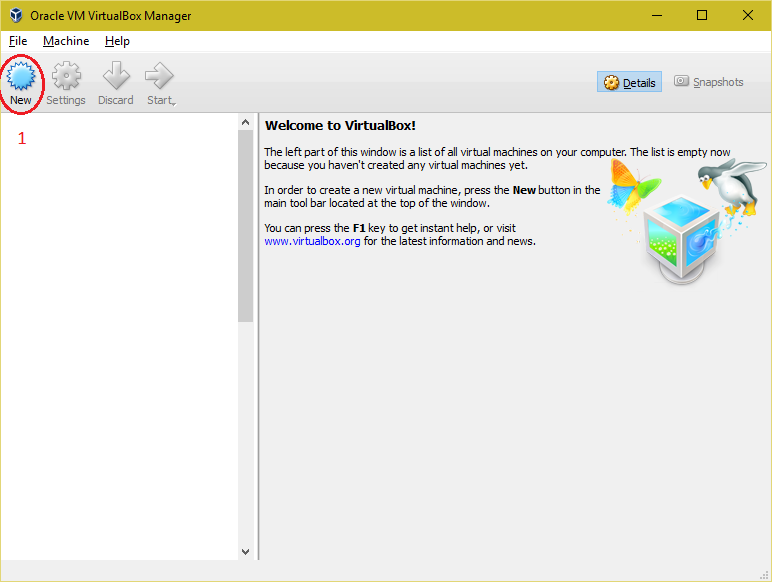
A screenshot of a cell phone

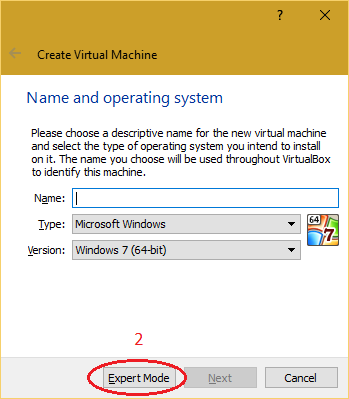
Description generated with very high confidence



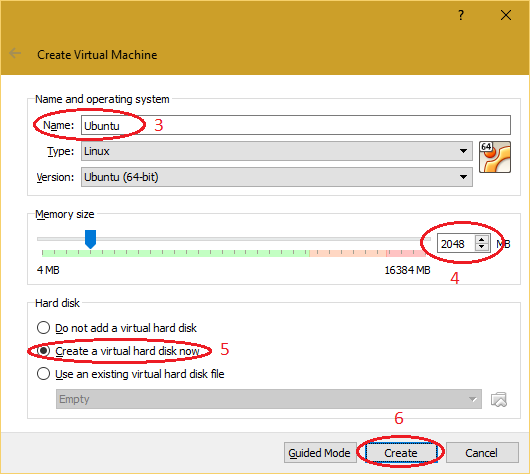


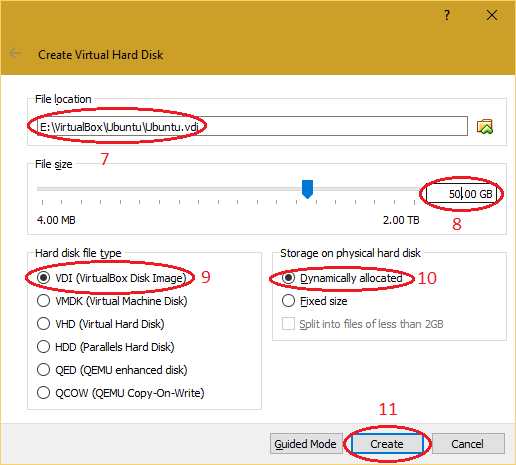
1. Click on "New" button and select "Expert Mode”

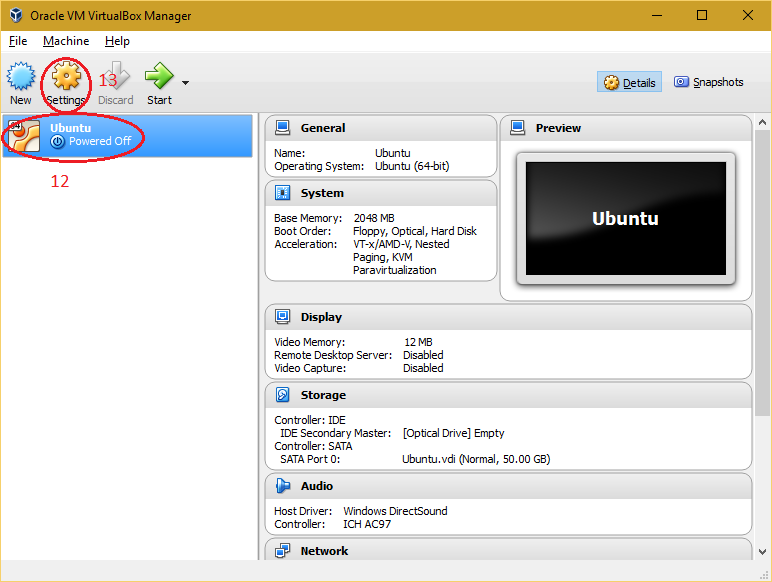




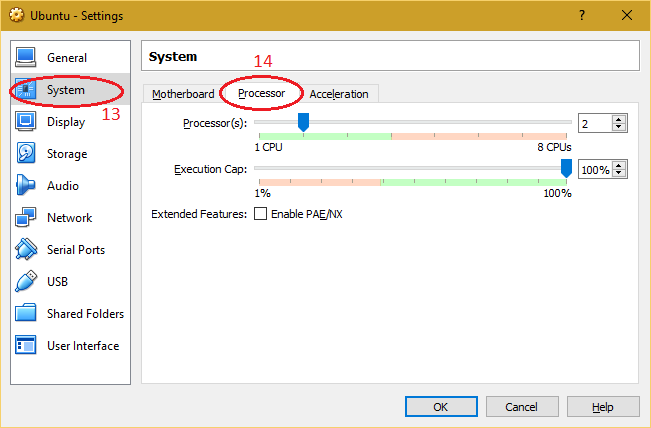
1. Provide the name and operating system information for virtual machine.



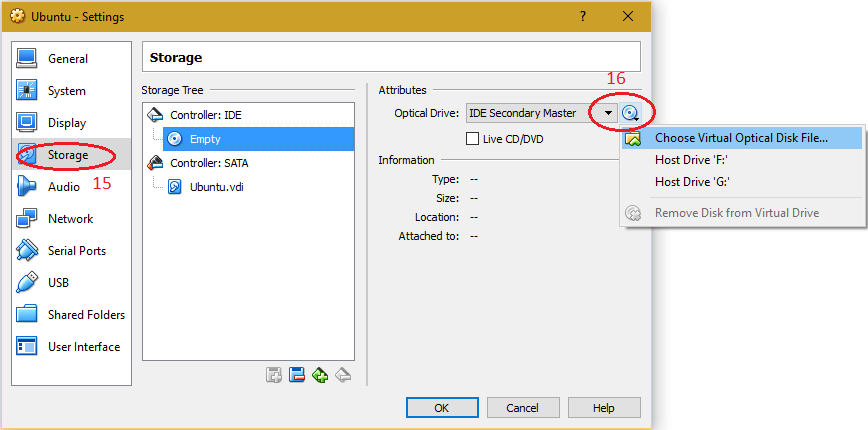
1. Select the path for the virtual hard disk and click on "Create" button. 
2. Select the virtual machine from the virtual box manager and click on "Settings" button.

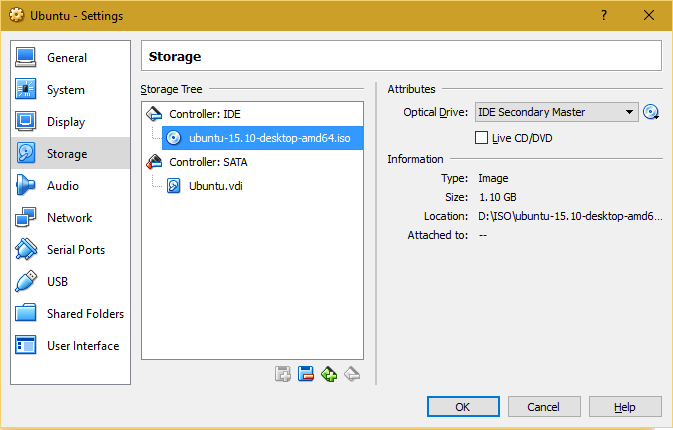


1. Select "System" and navigate to "Processor" tab to adjust number of processor of virtual machine for better performance.

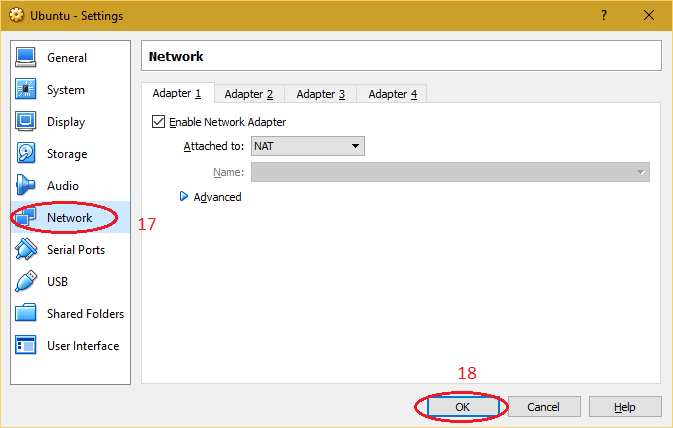


1. Select "Storage" and choose the installation media of Operating System (ISO/CD/DVD).

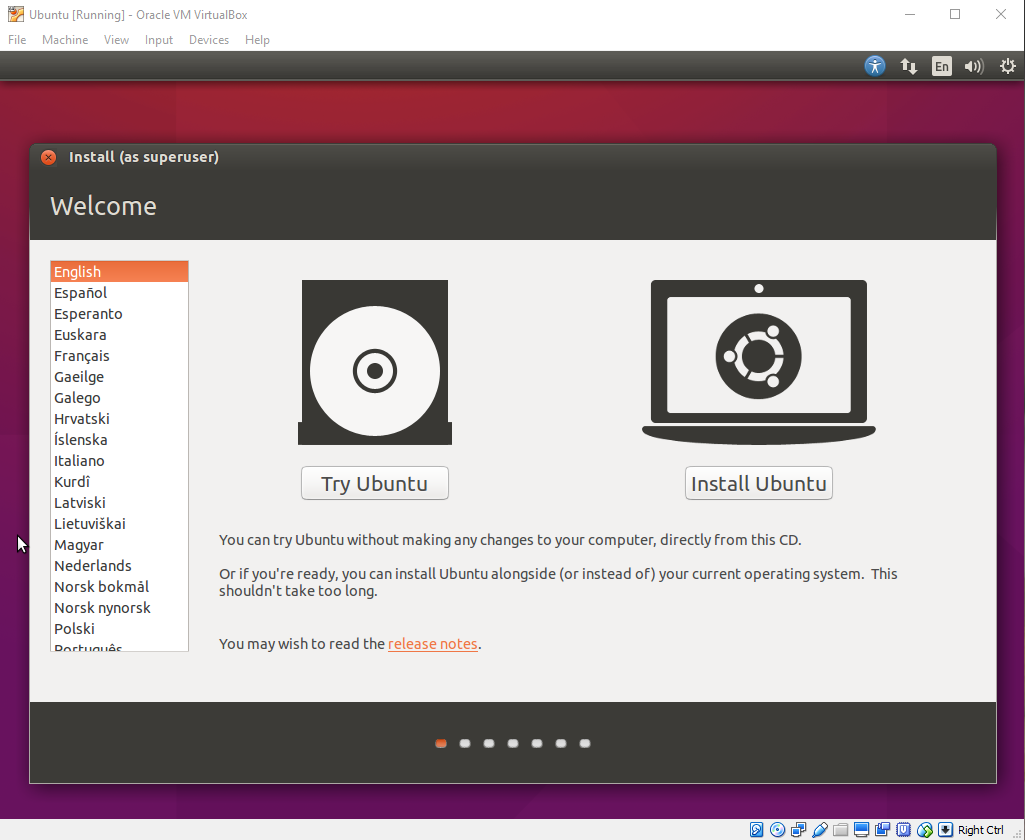




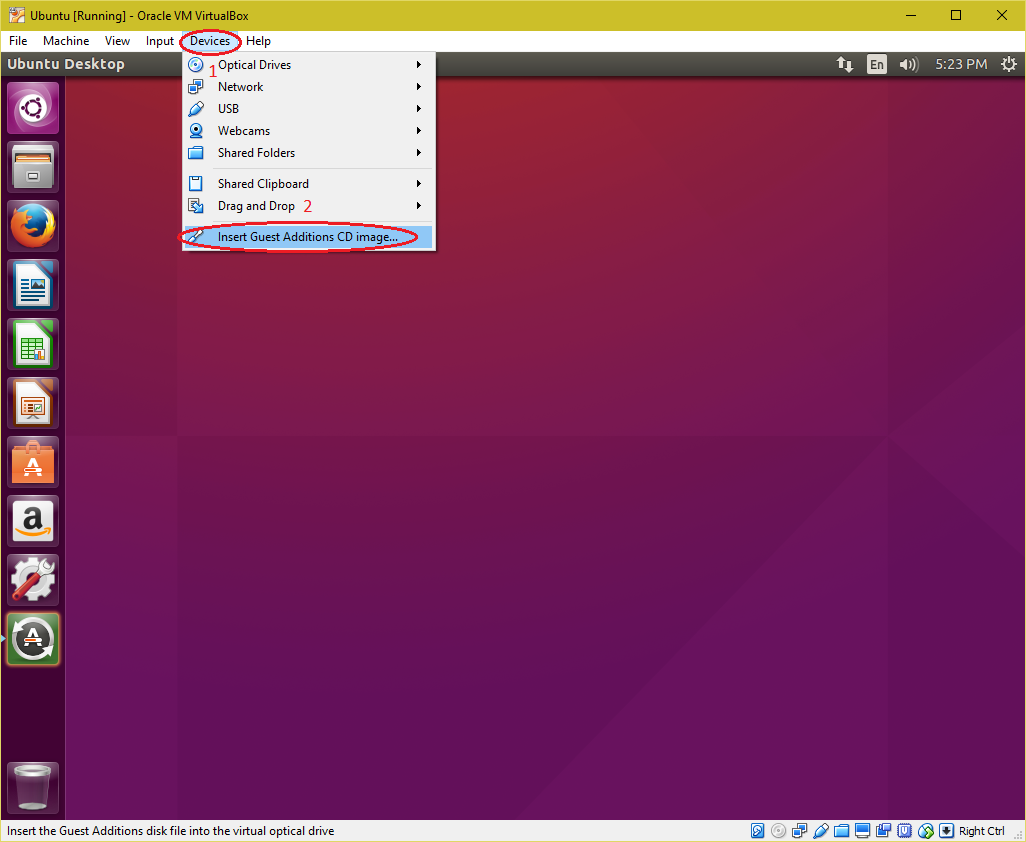
1. Select "Network" to make changes required for network setting of virtual machine and click on "OK

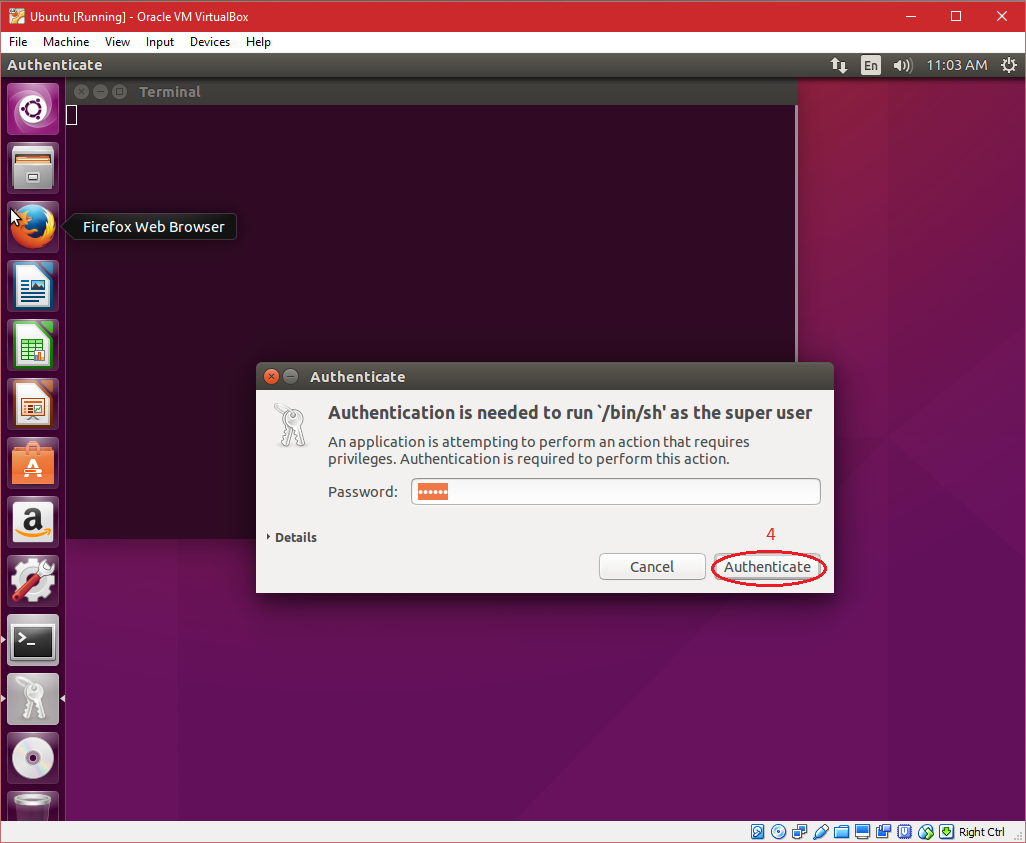


1. Select the created virtual machine and click on "Start" button
2. Proceed with the installation of operating system in virtual machine



1. Then after finish of installation open the installed OS.





1. The number of VMs you can be utilized is limited only by your system’s resources.
2. Since each VM requires a certain amount of CPU, RAM, memory, and other resources, your physical system’s configuration determines the number of VMs you can install on it
3. However, remember virtualization adds some overhead to your system, so it may be slower than what you may experience on real hardware. For this reason, it’s not a good idea to install multiple apps or games that demand a ton of resources.

**Result**

Thus procedure to run the virtual machine of different configuration and how many virtual machine can be utilized at particular time is successfully done and checked.

|  |  |
| --- | --- |
| **EX.NO : 3** | FIND PROCEDURE TO ATTACH VIRTUAL BOX TO VIRTUAL MACHINE AND CHECK WHETHER IT HOLDS THE DATA EVEN AFTER THE RELEASE OF THE VIRTUAL BOX |
| **DATE:** |

**Aim**

To find procedure to attach virtual box to virtual machine and to check whether it holds the data even after the release of the virtual box

**Procedure**

1. Download the appropriate package for your distribution which you need
2. The use the command

sudo dpkg -i virtualbox5.0\_5.0.20\_Ubuntu\_wily\_i386.deb

1. The installer will also try to build kernel modules suitable for the current running kernel. If the build process is not successful you will be shown a warning and the package will be left unconfigured. Please have a look at /var/log/vboxinstall.log to find out why the compilation failed
2. You may have to install the appropriate Linux kernel After correcting any problems, do

sudo rcv boxdrv setup

1. This will start a second attempt to build the module.

sudo make install

1. The VirtualBox kernel module needs a device node to operate. The above make command will tell you how to create the device node, depending on your Linux system. The procedure is slightly different for a classical Linux setup with a /dev directory, a system with the now deprecated devfs and a modern Linux system with udev
2. Next, you will have to install the system initialization script for the kernel module:

cp /opt/VirtualBox/vboxdrv.sh/sbin/rcvboxdrv

mkdir /etc/vbox

echo INSTALL\_DIR=/opt/VirtualBox> /etc/vbox/vbox.cfg

and, for convenience, create the following symbolic links:

ln-sf /opt/VirtualBox/VBox.sh /usr/bin/VirtualBox

ln -sf /opt/VirtualBox/VBox.sh /usr/bin/VBoxManage

ln -sf /opt/VirtualBox/VBox.sh /usr/bin/VBoxHeadless

ln -sf /opt/VirtualBox/VBox.sh /usr/bin/VBoxSDL

**Result**

Thus the procedure to attach virtual box to virtual machine and the data holds even after the release of the virtual box is checked and executed successfully

|  |  |
| --- | --- |
| **EX.NO : 4** | INSTALL A C COMPILER IN VIRTUAL MACHINE AND EXECUTE A SAMPLE PROGRAM |
| **DATE:** |

**Aim :**

To install a C compiler in the virtual machine and execute a sample program.

**Procedure :**

1. Download and install virtual box manager and open the virtual box manager
2. Open the required virtual machine in Ubuntu
3. Navigate to the terminal of the machine
4. Download the source file for installing compiler from any source
5. While installing the compiler make sure that you are superuser to execute command

To become super user

use **sudo su** command

1. Then follow as

tar xzf gcc-4.6.2.tar.gz

cd gcc-4.6.2

./contrib/download-prerequisites

cd ..

mkdir objdir

$PUD/../gcc-4.6.2/configure—prefix=$HOME/gcc-4.6.2—enable-languages=c,c++, -disable= multilib

make

make install

1. After installing, execute the program

**Program:**

#include<stdio.h>

void main()

{

print(“Hello World”);

}

1. To run and execute, gcc first.c -o first

./first

**Output**

Hello World

**Result**

Thus installation of a C compiler in the virtual machine and a sample program is executed successfully.

|  |  |
| --- | --- |
| **EX.NO : 5** | WRITE A WORD COUNT PROGRAM TO DEMOSTRATE THE USE OF MAP AND REDUCE TASK |
| **DATE:** |

**Aim :**

To count the number of occurrences of each word in a given file using Map and Reduce task.

**Procedure :**

Step 1 : Start the program.

Step 2 : Open Eclipse and create a new Java File and save it as “WordCount.java”

Step 3 : Open the WordCount java file.

Step 4 : Create a Map class and Reduce class to map the word sets and reduce words into word groups.

Step 5 : In the main method, create a Job for both Map and Reduce classes.

Step 6 : Provide the input file source to Job using command line arguments.

Step 7 : Open command prompt and navigate to sbin folder in Hadoop, then type start-all and visit localhost:50070/

Step 8 : In the Utilities, go to File system and open the output file to see the output.

Step 9 : Stop the program.

**Program :**

import java.io.IOException;

import java.util.StringTokenizer;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.LongWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Mapper;

import org.apache.hadoop.mapreduce.Reducer;

import org.apache.hadoop.conf.Configuration;

import org.apache.hadoop.mapreduce.Job;

import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;

import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;

import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;

import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

import org.apache.hadoop.fs.Path;

public class WordCount

{

public static class Map extends Mapper<LongWritable,Text,Text,IntWritable> {

public void map(LongWritable key, Text value,Context context) throws IOException,InterruptedException{

String line = value.toString();

StringTokenizer tokenizer = new StringTokenizer(line);

while (tokenizer.hasMoreTokens()) {

value.set(tokenizer.nextToken());

context.write(value, new IntWritable(1));

}

}

}

public static class Reduce extends Reducer<Text,IntWritable,Text,IntWritable> {

public void reduce(Text key, Iterable<IntWritable> values,Context context) throws IOException,InterruptedException {

int sum=0;

for(IntWritable x: values)

{

sum+=x.get();

}

context.write(key, new IntWritable(sum));

}

}

public static void main(String[] args) throws Exception {

Configuration conf= new Configuration();

Job job = new Job(conf,"My Word Count Program");

job.setJarByClass(WordCount.class);

job.setMapperClass(Map.class);

job.setReducerClass(Reduce.class);

job.setOutputKeyClass(Text.class);

job.setOutputValueClass(IntWritable.class);

job.setInputFormatClass(TextInputFormat.class);

job.setOutputFormatClass(TextOutputFormat.class);

Path outputPath = new Path(args[1]);

FileInputFormat.addInputPath(job, new Path(args[0]));

FileOutputFormat.setOutputPath(job, new Path(args[1]));

outputPath.getFileSystem(conf).delete(outputPath);

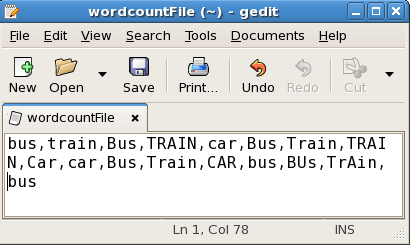
//exiting the job only if the flag value becomes false

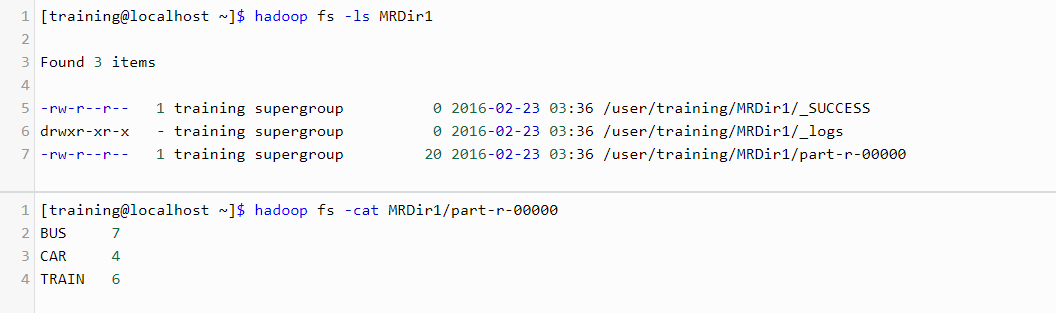
System.exit(job.waitForCompletion(true) ? 0 : 1);

}

}

**OUTPUT**

****

****

**Result:**

Thus the count of occurrence of words in the given file using Hadoop map and reduce task is executed and verified successfully.

|  |  |
| --- | --- |
| **EX.NO : 6** | INSTALLATION AND CONFIGURATION OF HADOOP |
| **DATE:** |

**Aim**

To install and configure Hadoop.

**Procedure**

Step 1 : Extract the hadoop-2.8.0.tar.gz package into local directory.

Step 2 : Change to the extracted directory.

Step 3 : To set the environmental variables for hadoop and java, Open the system settings.

Step 4 : In the Environmental variables, create JAVA\_HOME and HADOOP\_HOME as system variables.

Step 5 : Then add Java and hadoop bin folder to PATH variable as C:\Progra~1\Java\bin and C:\hadoop-2.8.0\bin.

Step 6 : In Hadoop-env.cmd, change the JAVA\_HOME variable to bin folder of java.

Step 7 : To verify the assignment of environmental variables, type,

*java -version* and *javac -version* to display the versions of installed.

Step 8 : To format the namenode, after installation, type

*hadoop namenode format.*

Step 9 : type *hadoop* to verify the installation of hadoop.

**Result:**

Thus hadoop is installed and configured in the system successfully.

|  |  |
| --- | --- |
| **EX.NO : 7** | INSTALLATION AND RUN R PROGRAMMING THEN USE PROGRAMMING STATEMENTS (IF STATEMENT, FOR LOOP AND OWN FUNCTION) |
| **DATE:** |

**Aim**

To install R Studio and R Program and execute a basic R Program with If, For, User Defined Function.

**Procedure**

Step 1 : Install R Programming.

Step 2 : Install R Studio.

Step 3 : Open the R Studio and navigate to File New File and create a new R File and save it.

Step 4 : Write a R Program with a If Block, For Loop and a User Defined Function in it.

Step 5 : To Run the program, Select the entire program and click the Run button on the top of the editor.

Step 6 : The Output of the current program is shown in the console Window below.

**Program**

* **Basic If Program:**

x <- 5

if(x > 0){

print("Positive number")

}

* **Basic For Loop:**

vec <- c(2,5,3,9,8,11,6)

count <- 0

for (i in vec) {

if(i %% 2 == 0)

count = count+1

}

print(count)

* **User Defined Function:**

fun <-function(a,b){

return (a+b)

}

print(fun(5,6))

**OUTPUT**

Positive number

3

11

**Result :**

Thus the R Program with the given basics components is successfully executed and verified.

|  |  |
| --- | --- |
| **EX.NO : 8** | IMPLEMENT MATRIX MULTIPLICATION WITH R PROGRAM |
| **DATE:** |

**Aim :**

To implement matrix multiplication with R Programming.

**Procedure :**

Step 1 : Start the program.

Step 2 : Open R Studio.

Step 3 : Create two matrices A and B using matrix method.

Step 4 : Perform matrix multiplication and store the result in another r matrix.

Step 5 : Print the result matrix.

Step 6 : Stop the program

**Program**

A <- matrix(c(2,2,1,2,3,1),2,3)

B <- matrix(c(3,1,2,2,1,3,3,2,1),3,3)

print(“A = ”,A)

print(“B = ”,B)

C <- A %\*% B

print(C)

**Output**

13 6 11

15 7 14

**Result :**

Thus the program for matrix multiplication is executed and verified successfully.

|  |  |
| --- | --- |
| **EX.NO : 9** | INSTALL AND RUN PIG THEN WRITE PIG LATIN SCRIPTS TO SORT, GROUP , JOIN, JOIN AND FILTER YOUR DATA |
| **DATE:** |

**Aim :**

To install pig and execute latin scripts to sort, group, join, project, and filter your data.

**Procedure :**

Step 1 : Gather the pig install release from any source.

Step 2 : Extract the latest release of pig to your local directory.

Step 3 : Change the working directory to the extracted folder.

Step 4 : Set the environmental variable for PIG\_HOME to the root folder of pig.

Step 5 : Check whether JAVA\_HOME and HADOOP\_HOME is set as environmental variables.

Step 6 : To verify the installation, open command prompt and type *pig*  in it.so it displays the grunt console.

Consider the dataset, as dataset.txt

001,Rajiv,Reddy,21,9848022337,Hyderabad

002,siddarth,Battacharya,22,9848022338,Kolkata

003,Rajesh,Khanna,22,9848022339,Delhi

004,Preethi,Agarwal,21,9848022330,Pune

005,Trupthi,Mohanthy,23,9848022336,Bhuwaneshwar

006,Archana,Mishra,23,9848022335,Chennai

007,Komal,Nayak,24,9848022334,trivendram

008,Bharathi,Nambiayar,24,9848022333,Chennai

Step 7 : To sort the data,

**grunt >** student\_details = LOAD 'hdfs://localhost:9000/pig\_data/dataset.txt' USING PigStorage(',') as (id:int, firstname:chararray, lastname:chararray,age:int, phone:chararray, city:chararray);

**grunt >** order\_by\_data = ORDER student\_details BY age DESC;

**grunt >** Dump order by\_data;

**Output**

(8,Bharathi,Nambiayar,24,9848022333,Chennai)

(7,Komal,Nayak,24,9848022334,trivendram)

(6,Archana,Mishra,23,9848022335,Chennai)

(5,Trupthi,Mohanthy,23,9848022336,Bhuwaneshwar)

(3,Rajesh,Khanna,22,9848022339,Delhi)

(2,siddarth,Battacharya,22,9848022338,Kolkata)

(4,Preethi,Agarwal,21,9848022330,Pune)

(1,Rajiv,Reddy,21,9848022337,Hyderabad)

Step 8 : To group the data,

**grunt >** student\_details = LOAD 'hdfs://localhost:9000/pig\_data/dataset.txt' USING PigStorage(',') as (id:int, firstname:chararray, lastname:chararray,age:int, phone:chararray, city:chararray);

**grunt >** group\_data = GROUP student\_details by age;

**grunt >** grunt> Dump group\_data;

**Output**

(21,{(4,Preethi,Agarwal,21,9848022330,Pune),(1,Rajiv,Reddy,21,9848022337,Hydera bad)})

(22,{(3,Rajesh,Khanna,22,9848022339,Delhi),(2,siddarth,Battacharya,22,984802233 8,Kolkata)})

(23,{(6,Archana,Mishra,23,9848022335,Chennai),(5,Trupthi,Mohanthy,23,9848022336 ,Bhuwaneshwar)})

(24,{(8,Bharathi,Nambiayar,24,9848022333,Chennai),(7,Komal,Nayak,24,9848022334, trivendram)})

Consider the dataset, as dataset1.txt

1,Ramesh,32,Ahmedabad,2000.00

2,Khilan,25,Delhi,1500.00

3,kaushik,23,Kota,2000.00

4,Chaitali,25,Mumbai,6500.00

5,Hardik,27,Bhopal,8500.00

6,Komal,22,MP,4500.00

7,Muffy,24,Indore,10000.00

Consider the dataset, as dataset2.txt

102,2009-10-08 00:00:00,3,3000

100,2009-10-08 00:00:00,3,1500

101,2009-11-20 00:00:00,2,1560

103,2008-05-20 00:00:00,4,2060

Step 9 : To join the data, consider the dataset,

**grunt >** customer\_one = LOAD 'hdfs://localhost:9000/pig\_data/dataset1.txt' USING PigStorage(',') as (id:int, name:chararray, age:int, address:chararray, salary:int);

**grunt >** customer\_two= LOAD 'hdfs://localhost:9000/pig\_data/dataset2.txt' USING PigStorage(',') as (oid:int, date:chararray, customer\_id:int, amount:int);

**grunt >** join\_data = JOIN customer\_one BY id, customer\_two BY id;

**grunt >** Dump join\_data;

**Output**

(1,Ramesh,32,Ahmedabad,2000,1,Ramesh,32,Ahmedabad,2000)

(2,Khilan,25,Delhi,1500,2,Khilan,25,Delhi,1500)

(3,kaushik,23,Kota,2000,3,kaushik,23,Kota,2000)

(4,Chaitali,25,Mumbai,6500,4,Chaitali,25,Mumbai,6500)

(5,Hardik,27,Bhopal,8500,5,Hardik,27,Bhopal,8500)

(6,Komal,22,MP,4500,6,Komal,22,MP,4500)

(7,Muffy,24,Indore,10000,7,Muffy,24,Indore,10000)

Step 10 : To filter the data, consider the dataset,

**grunt >** student\_details = LOAD 'hdfs://localhost:9000/pig\_data/dataset.txt' USING PigStorage(',') as (id:int, firstname:chararray, lastname:chararray,age:int, phone:chararray, city:chararray);

**grunt >** filter\_data = FILTER student\_details BY city == 'Chennai';

**grunt >** Dump filter\_data;

**Output**

(6,Archana,Mishra,23,9848022335,Chennai)

(8,Bharathi,Nambiayar,24,9848022333,Chennai)

**Result :**

Thus pig is installed and pig latin scripts to perform sort, join, group, filter the data is executed and verified successfully.

|  |  |
| --- | --- |
| **EX.NO : 10** | INSTAL AND RUN HIVE THEN USE HIVE TO CREATE, ALTER AND DROP DATABASE, TABLES, VIEWS, FUNCTIONS AND INDEXES |
| **DATE:** |

**Aim :**

To install Hive and use Hive to create, alter and drop databases, tables, views, functions and indexes.

**Procedure :**

Step 1 : Gather the hive install release from any source.

Step 2 : Extract the latest release of hive to your local directory.

Step 3 : Change the working directory to the extracted folder.

Step 4 : Set the environmental variable for HIVE\_HOME to the root folder of pig.

Step 5 : Check whether JAVA\_HOME and HADOOP\_HOME is set as environmental variables.

Step 6 : To verify the installation, open command prompt and type *hive --version* in it, to display the version of installed hive.

Step 7 : To create hive directories within HDFS, type,

* *hdfs dfs -mkdir -p /user/hive/warehouse*
* *hdfs dfs -mkdir /tmp*

To provide write permissions to the group,

* *hdfs dfs -chmod g+w /user/hive/warehouse*
* *hdfs dfs -chmod g+w /tmp*

Step 8 : Then add environmental variable HIVE\_CONF\_DIR to conf folder of hive.

Step 9 : To initialize the derby database,

* *bin/schematool -initSchema -dbType derby*

Step 9 : To launch hive, type *hive* in command prompt.

Step 10 : To create a database,

**hive >** *CREATE DATABASE [IF NOT EXISTS] userdb;*

**Output**

Database userdb created successfully.

Step 11 : To create a table,

**hive >** *CREATE TABLE IF NOT EXISTS employee ( eid int, name String,*

*salary String, destination String)*

*COMMENT ‘Employee details’*

*ROW FORMAT DELIMITED*

*FIELDS TERMINATED BY ‘\t’*

*LINES TERMINATED BY ‘\n’*

*STORED AS TEXTFILE;*

**Output**

Table employee created.

Step 12 : To alter the table,

**hive >** *ALTER TABLE employee RENAME TO emp;*

**Output**

Table renamed successfully

Step 13 : To drop the table,

**hive >** *DROP TABLE IF EXISTS employee;*

**Output**

Drop table successful

Step 14 : To drop the database,

**hive >** *DROP DATABASE IF EXISTS userdb;*

**Output**

Drop userdb database successful

Step 15 : To create and drop a view,

**hive >** *CREATE VIEW emp\_view AS*

*SELECT \* FROM employee WHERE salary > 30000;*

**Output**

View emp\_view created

**hive >** *DROP VIEW emp\_view;*

**Output**

Drop emp\_view view successful

Step 16 : To create and drop index,

**hive >** *CREATE INDEX salary\_index ON TABLE employee(salary)*

*AS org.apache.hadoop.hive.ql.index.compact.CompactIndexHandler';*

**Output**

Index salary\_index created

**hive >** *DROP INDEX salary\_index ON employee;*

**Output**

Drop salary\_index index successfully

Step 17 : To use a functions, add them with select query.

**hive >** *SELECT max(salary) from employee;*

**Output**

70000

**hive >** *SELECT count(salary) from employee;*

**Output**

6

**Result :**

Thus hive is installed and operation on database and tables are executed and verified successfully.